



[Symmetric Key Services Markup Language (SKSML) Version 1.0 Normative DRAFT 6.0]

OASIS Technical Committee DRAFT

24 June 2008

Specification URIs:

This Version:

- <http://docs.oasis-open.org/ekmi/1.0/SKSML-1.0-Specification.html>
- <http://docs.oasis-open.org/ekmi/1.0/SKSML-1.0-Specification.odt>
- <http://docs.oasis-open.org/ekmi/1.0/SKSML-1.0-Specification.pdf>

Previous Version:

None

Latest Version:

Same as "This Version"

Latest Approved Version:

None

Technical Committee:

OASIS Enterprise Key Management Infrastructure (EKMI) TC

Chair(s):

Arshad Noor, StrongAuth, Inc. (arshad.noor@strongauth.com)

Editor(s):

Allen Schaaf (netsecurity@sound-by-design.com)

Related Work:

This specification replaces or supercedes:

- None

This specification is related to:

- Advanced Encryption Standard (AES) - NIST FIPS 197 - <http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf>
- Simple Object Access Protocol (SOAP) - W3C Recommendation 08 May 2000. <http://www.w3.org/TR/soap/>
- XML Encryption - W3C Recommendation 10 Dec 2002. <http://www.w3.org/TR/xmlenc-core/>

- 33 • XML Signature - W3C Recommendation 12 Feb 2002. <http://www.w3.org/TR/xmlsig-core/>
34 • Web Services Security - SOAP Message Security 1.0 - OASIS Standard 200401, March 2004 -
35 <http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-soap-message-security-1.0.pdf>

36 **Declared XML Namespace(s):**

37 <http://docs.oasis-open.org/ekmi/2008/01>

38 **Abstract:**

39 This normative specification defines the first (1.0) version of the Symmetric Key Services Markup
40 Language (SKSML).

41 **Status:**

42 This document was last revised by the EKMI TC as of the above date. The level of approval is also
43 listed above. Check the "Latest Version" or "Latest Approved Version" location noted above for possible
44 later revisions of this document.

45 Technical Committee members should send comments on this specification to the Technical
46 Committee's email list. Others should send comments to the Technical Committee by using the "Send A
47 Comment" button on the Technical Committee's web page at [http://www.oasis-](http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=ekmi)
48 [open.org/committees/tc_home.php?wg_abbrev=ekmi](http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=ekmi)

49 For information on whether any patents have been disclosed that may be essential to implementing this
50 specification, and any offers of patent licensing terms, please refer to the Intellectual Property Rights
51 section of the Technical Committee web page (<http://www.oasis-open.org/committees/ekmi/ipr.php>).

52 The non-normative errata page for this specification is located at [http://www.oasis-open.org/committees/](http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=ekmi/)
53 [tc_home.php?wg_abbrev=ekmi/](http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=ekmi/).

Notices

54

55 Copyright © OASIS® 2008. All Rights Reserved.

56 All capitalized terms in the following text have the meanings assigned to them in the OASIS Intellectual Property
57 Rights Policy (the "OASIS IPR Policy"). The full Policy may be found at the OASIS website.

58 This document and translations of it may be copied and furnished to others, and derivative works that comment
59 on or otherwise explain it or assist in its implementation may be prepared, copied, published, and distributed, in
60 whole or in part, without restriction of any kind, provided that the above copyright notice and this section are
61 included on all such copies and derivative works. However, this document itself may not be modified in any way,
62 including by removing the copyright notice or references to OASIS, except as needed for the purpose of
63 developing any document or deliverable produced by an OASIS Technical Committee (in which case the rules
64 applicable to copyrights, as set forth in the OASIS IPR Policy, must be followed) or as required to translate it into
65 languages other than English.

66 The limited permissions granted above are perpetual and will not be revoked by OASIS or its successors or
67 assigns.

68 This document and the information contained herein is provided on an "AS IS" basis and OASIS DISCLAIMS ALL
69 WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE
70 OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY OWNERSHIP RIGHTS OR ANY IMPLIED
71 WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

72 OASIS requests that any OASIS Party or any other party that believes it has patent claims that would necessarily
73 be infringed by implementations of this OASIS Committee Specification or OASIS Standard, to notify OASIS TC
74 Administrator and provide an indication of its willingness to grant patent licenses to such patent claims in a
75 manner consistent with the IPR Mode of the OASIS Technical Committee that produced this specification.

76 OASIS invites any party to contact the OASIS TC Administrator if it is aware of a claim of ownership of any
77 patent claims that would necessarily be infringed by implementations of this specification by a patent holder that
78 is not willing to provide a license to such patent claims in a manner consistent with the IPR Mode of the OASIS
79 Technical Committee that produced this specification. OASIS may include such claims on its website, but
80 disclaims any obligation to do so.

81 OASIS takes no position regarding the validity or scope of any intellectual property or other rights that might be
82 claimed to pertain to the implementation or use of the technology described in this document or the extent to
83 which any license under such rights might or might not be available; neither does it represent that it has made
84 any effort to identify any such rights. Information on OASIS' procedures with respect to rights in any document or
85 deliverable produced by an OASIS Technical Committee can be found on the OASIS website. Copies of claims of
86 rights made available for publication and any assurances of licenses to be made available, or the result of an
87 attempt made to obtain a general license or permission for the use of such proprietary rights by implementers or
88 users of this OASIS Committee Specification or OASIS Standard, can be obtained from the OASIS TC
89 Administrator. OASIS makes no representation that any information or list of intellectual property rights will at
90 any time be complete, or that any claims in such list are, in fact, Essential Claims.

91 The names "OASIS", [insert specific trademarked names, abbreviations, etc. here] are trademarks of OASIS, the
92 owner and developer of this specification, and should be used only to refer to the organization and its official
93 outputs. OASIS welcomes reference to, and implementation and use of, specifications, while reserving the right
94 to enforce its marks against misleading uses. Please see <http://www.oasis-open.org/who/trademark.php> for
95 above guidance.

96

Table of Contents

98	1 Introduction.....	5
99	1.1 Terminology.....	5
100	1.2 Glossary.....	5
101	1.3 Normative References.....	6
102	2 Specification.....	7
103	2.1 Element <SymkeyRequest>.....	7
104	2.2 Element <GlobalKeyID>.....	9
105	2.3 Element <KeyClasses> and <KeyClass>.....	10
106	2.4 Element <SymkeyResponse>.....	12
107	2.5 Element <Symkey>.....	13
108	2.6 Element <SymkeyError>.....	15
109	2.7 Element <KeyUsePolicy>.....	17
110	2.8 Type TwoPartIDType.....	19
111	2.9 Element <KeyAlgorithm>.....	20
112	2.10 Element <KeySize>.....	21
113	2.11 Element <Status>.....	22
114	2.12 Element <Permissions>.....	23
115	2.13 Element <PermittedApplications> and <PermittedApplication>.....	29
116	2.14 Element <PermittedDates> and <PermittedDate>.....	32
117	2.15 Element <PermittedDays> and <PermittedDay>.....	34
118	2.16 Element <PermittedDuration>.....	35
119	2.17 Element <PermittedLevels> and <PermittedLevel>.....	36
120	2.18 Element <PermittedLocations> and <PermittedLocation>.....	38
121	2.19 Element <PermittedNumberOfTransactions>.....	40
122	2.20 Element <PermittedTimes> and <PermittedTime>.....	41
123	2.21 Element <PermittedUses> and <PermittedUse>.....	43
124	2.22 Element <KeyCachePolicyRequest>.....	44
125	2.23 Element <KeyCachePolicyResponse>.....	45
126	2.24 Element <KeyCachePolicy>.....	45
127	2.25 Type KeyCacheDetailType.....	48
128		

129 1 Introduction

130 This document presents the specification for the Symmetric Key Services Markup Language (SKSML), a protocol
131 by which applications may request and receive symmetric key-management services, securely, over networks or
132 other mechanisms as may be selected by implementers. All text is normative unless otherwise indicated.

133 1.1 Terminology

134 The keywords "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT",
135 "RECOMMENDED", "MAY", and "OPTIONAL" in this specification are to be interpreted as described in IETF
136 RFC 2119 .

137 1.2 Glossary

138 **3DES** – Triple Data Encryption Standard

139 **AES** – Advanced Encryption Standard

140 **Base64** – An encoding scheme for representing data

141 **Ciphertext** – Encrypted data

142 **Cryptographic module** – A software library or hardware module dedicated to performing cryptographic
143 operations

144 **DES** – Data Encryption Standard

145 **DID or Domain ID** – Domain Identifier; the unique **PEN** assigned to an implementation of an **SKMS** (Symmetric
146 Key Management System) within an enterprise

147 **GKID or Global Key ID** – Global Key Identifier; the unique identifier assigned to every symmetric encryption key
148 within an **SKMS**. It is the concatenation of the **DID-SID-KID**

149 **Initialization Vector or IV** – A block of bits required to encrypt/decrypt the first block of data when used with a
150 particular mode of cryptographic operations

151 **KeyCachePolicy** – The collection of rules that defines how a symmetric encryption key may be cached by a
152 client implementation

153 **KID or Key ID** – Key Identifier; the unique integer assigned to every symmetric encryption key generated within a
154 specific **SKS** (Symmetric Key Services) server within an **SKMS** (Symmetric Key Management System)

155 **KeyUsePolicy** – The collection of rules that defines how a symmetric encryption key may be used by an
156 application

157 **PEN** – Private Enterprise Number; the unique integer assigned by IANA to any organization that requests such a
158 number

159 **PII** – Personally Identifiable Information, such as credit card numbers, social security numbers, bank account
160 numbers, drivers license numbers, etc.

161 **Plaintext** – Unencrypted data

162 **SHA** – Secure Hashing Algorithm

163 **SHA-1** – Secure Hashing Algorithm with a resultant size of 160-bits

164 **SHA-256** – Secure Hashing Algorithm with a resultant size of 256-bits

165 **SHA-384** – Secure Hashing Algorithm with a resultant size of 384-bits

166 **SHA-512** – Secure Hashing Algorithm with a resultant size of 512-bits

167 **SID** or **Server ID** – Server Identifier; the unique integer assigned to every **SKS** server within an enterprise's **SKMS**

168 **SKCL** – Symmetric Key Client Library; a software library that supports the **SKSML** protocol

169 **SKMS** – Symmetric Key Management System; a collection of hardware and software providing symmetric
170 encryption key-management services

171 **SKS** – Symmetric Key Services; a server that provides symmetric key management services over a network or
172 other mechanism selected by implementers

173 **SKSML** – Symmetric Key Services Markup Language; an XML-based protocol to request and receive symmetric
174 encryption key-management services

175 **SOAP** – Simple Object Access Protocol

176 **SOAP Body** – The content part of a SOAP message

177 **SOAP Envelope** – The SOAP message consisting of a SOAP Header and a SOAP Body, conforming to the
178 SOAP protocol standard.

179 **SOAP Error** – A SOAP error message response to a SOAP request

180 **SOAP Header** – The header part of a SOAP message containing meta-information about the message, including
181 security-related objects

182 **Symkey** - A symmetric encryption key

183 **unbounded** – A parameter used with the “maxOccurs” attribute to indicate an unlimited number

184 **XMLEncryption** – Encrypted content represented in eXtensible Markup Language that conforms to the World
185 Wide Web Consortium's XML Encryption standard

186 **XMLSignature** – A digital signature represented in eXtensible Markup Language that conforms to the World
187 Wide Web Consortium's XML Signature standard

188 **1.3 Normative References**

189 **[AES]** Advanced Encryption Standard
190 NIST FIPS 197. <http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf>

191 **[RFC 2119]** S. Bradner. *Key words for use in RFCs to Indicate Requirement Levels*.
192 IETF RFC 2119, March 1997. <http://www.ietf.org/rfc/rfc2119.txt>

193 **[SOAP]** Simple Object Access Protocol 1.1
194 W3C Recommendation 08 May 2000. <http://www.w3.org/TR/soap/>

195 **[XMLEncryption]** XML Encryption Syntax and Processing
196 W3C Recommendation 10 Dec 2002. <http://www.w3.org/TR/xmlenc-core/>

197 **[XMLSignature]** XML Signature Syntax and Processing
198 W3C Recommendation 12 Feb 2002. <http://www.w3.org/TR/xmlsig-core/>

199 **[WSS]** Web Services Security – SOAP Message Security 1.0
200 OASIS Standard 200401, March 2004
201 <http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-soap-message->
202 [security-1.0.pdf](http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-soap-message-security-1.0.pdf)

203 **[RFC 2578]** K. McCloghrie, et al. *Structure of Management Information Version 2 (SMIv2)*.
204 IETF RFC 2578, April 1999. <http://www.ietf.org/html/rfc2578>

205

206

2 Specification

2.1 Element <SymkeyRequest>

The <SymkeyRequest> element identifies one or more *GlobalKeyID*'s of symmetric encryption keys needed by the client application. The request may also specify one or more *KeyClass* elements for the requested key when the request is for a new symmetric key.

While it is a top-level element within this specification, a <SymkeyRequest> element MUST be enclosed within a *SOAP Body* element of a *SOAP Envelope* to conform to the security requirements of this specification. The *SOAP Header* of the *SOAP Envelope* MUST enclose a *Security* element conforming to [WSS] with a *ValueType* attribute containing the value <http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-x509-token-profile-1.0#X509v3>. The *Security* element must conform to all other requirements of the specified security profile in [WSS] to form a well-formed, secure message.

Schema Definition:

```
<xsd:element name="SymkeyRequest">
  <xsd:complexType>
    <xsd:sequence>
      <xsd:element
        name="GlobalKeyID"
        type="ekmi:GlobalKeyIDType"
        minOccurs="1"
        maxOccurs="unbounded">
      </xsd:element>
      <xsd:element
        name="KeyClasses"
        type="ekmi:KeyClassesType"
        minOccurs="0">
      </xsd:element>
    </xsd:sequence>
  </xsd:complexType>
</xsd:element>
```

The <SymkeyRequest> element consists of a sequence of two child elements:

1. <GlobalKeyID> [Required]

This element of type *GlobalKeyIDType*, identifies the unique global key identifier of the requested symmetric key within the target Symmetric Key Management System (SKMS) the client is communicating with. There MUST be at least one <GlobalKeyID> element in a <SymkeyRequest>, but there may be an unbounded (unlimited) number of <GlobalKeyID> elements specified.

The <GlobalKeyID> element is specified in Section 2.2.

2. <KeyClasses> [Optional]

This element of type *KeyClassesType*, when specified, identifies at least one <KeyClass> element, but may specify an unbounded (unlimited) number of <KeyClass> elements within the <KeyClasses> set. Client applications may request one or more symmetric keys conforming to one or more key classes required by the application. If the client application is authorized to receive keys conforming to such key classes, the SKS server will generate and supply them.

When more than one <GlobalKeyID> for a new symmetric key is specified in the request, there MAY be only one <KeyClass> element within the <KeyClasses> set.

When the client requires more than one new symmetric key, and each key is required to be of a different

257 key class, there MUST be only one <GlobalKeyID> element followed by as many <KeyClass>
258 elements inside the <KeyClasses> set, as needed by the client application.

259

260 When a client requires multiple symmetric keys of two or more key classes, the client MUST send
261 multiple requests to the SKS server. See examples 4 and 5 below in this section.

262

263 The <KeyClasses> and <KeyClass> elements are specified in Section 2.3.

264 Some examples of the <SymkeyRequest> element are as follows:

265 **Example 1 – A single new symmetric key request of a default key class:**

```
266 <ekmi:SymkeyRequest xmlns:ekmi="http://docs.oasis-open.org/ekmi/2008/01">  
267 <ekmi:GlobalKeyID>10514-0-0</ekmi:GlobalKeyID>  
268 </ekmi:SymkeyRequest>
```

269 **Example 2 – A request for three new symmetric keys of a default key class for each symmetric key:**

```
270 <ekmi:SymkeyRequest xmlns:ekmi="http://docs.oasis-open.org/ekmi/2008/01">  
271 <ekmi:GlobalKeyID>10514-0-0</ekmi:GlobalKeyID>  
272 <ekmi:GlobalKeyID>10514-0-0</ekmi:GlobalKeyID>  
273 <ekmi:GlobalKeyID>10514-0-0</ekmi:GlobalKeyID>  
274 </ekmi:SymkeyRequest>
```

275 **Example 3 – A request for a single new symmetric key of a specific key class:**

```
276 <ekmi:SymkeyRequest xmlns:ekmi="http://docs.oasis-open.org/ekmi/2008/01">  
277 <ekmi:GlobalKeyID>10514-0-0</ekmi:GlobalKeyID>  
278 <ekmi:KeyClasses>  
279 <ekmi:KeyClass>HR-Class</ekmi:KeyClass>  
280 </ekmi:KeyClasses>  
281 </ekmi:SymkeyRequest>
```

282 **Example 4 – A request for a two new symmetric keys with the same key class for each symmetric key:**

```
283 <ekmi:SymkeyRequest xmlns:ekmi="http://docs.oasis-open.org/ekmi/2008/01">  
284 <ekmi:GlobalKeyID>10514-0-0</ekmi:GlobalKeyID>  
285 <ekmi:GlobalKeyID>10514-0-0</ekmi:GlobalKeyID>  
286 <ekmi:KeyClasses>  
287 <ekmi:KeyClass>FIN-FX</ekmi:KeyClass>  
288 </ekmi:KeyClasses>  
289 </ekmi:SymkeyRequest>
```

290 **Example 5 – A request for a nine new symmetric keys of different key classes:**

```
291 <ekmi:SymkeyRequest xmlns:ekmi="http://docs.oasis-open.org/ekmi/2008/01">  
292 <ekmi:GlobalKeyID>10514-0-0</ekmi:GlobalKeyID>  
293 <ekmi:KeyClasses>  
294 <ekmi:KeyClass>EHR-CDC</ekmi:KeyClass>  
295 <ekmi:KeyClass>EHR-CRO</ekmi:KeyClass>  
296 <ekmi:KeyClass>EHR-DEF</ekmi:KeyClass>  
297 <ekmi:KeyClass>EHR-EMT</ekmi:KeyClass>  
298 <ekmi:KeyClass>EHR-HOS</ekmi:KeyClass>  
299 <ekmi:KeyClass>EHR-INS</ekmi:KeyClass>  
300 <ekmi:KeyClass>EHR-NUR</ekmi:KeyClass>  
301 <ekmi:KeyClass>EHR-PAT</ekmi:KeyClass>  
302 <ekmi:KeyClass>EHR-PHY</ekmi:KeyClass>  
303 </ekmi:KeyClasses>  
304 </ekmi:SymkeyRequest>
```


305 2.2 Element <GlobalKeyID>

306 The <GlobalKeyID> element is the unique identifier of a symmetric encryption key within an SKMS. Every
307 symmetric key generated by the SKS server MUST be assigned a unique <GlobalKeyID> as specified in this
308 section.

309 Schema Definition:

```
310 <xsd:simpleType name="GlobalKeyIDType">  
311   <xsd:restriction base="xsd:string">  
312     <xsd:minLength value="5"/>  
313     <xsd:maxLength value="62"/>  
314     <xsd:pattern value="[0-9]{1,20}-[0-9]{1,20}-[0-9]{1,20}"/>  
315     <xsd:whiteSpace value="collapse"/>  
316   </xsd:restriction>  
317 </xsd:simpleType>
```

318 The <GlobalKeyID> element is of the *GlobalKeyIDType*, and is a string identifier of a symmetric key
319 consisting of five parts concatenated together:

- 320 1. A positive integer identifying the *Domain ID*. The *DomainID* identifies the IANA-issued Private
321 Enterprise Number (PEN) as published at <http://www.iana.org/assignments/enterprise-numbers> and is
322 used by the SKS server to constrain the ownership of objects within the SKMS;
- 323 2. A literal hyphen ("-") without surrounding spaces;
- 324 3. A positive integer identifying the Server ID of the server that originally generated the key;
- 325 4. Another literal hyphen ("-") without surrounding spaces;
- 326 5. A positive integer identifying the Key ID;

327 Combined, the five components of this element make up a unique identifier for a symmetric key within the SKMS.
328 Since all enterprises are expected to use only the PENs assigned to them, technically the <GlobalKeyID> is
329 unique across the internet.

330 The *DomainID* part of the <GlobalKeyID> element MUST be a positive integer in the range of 0 (zero) to
331 18446744073709551615 (20-byte ASCII decimal).

332 When an SKMS manages the symmetric keys for a single enterprise, the *DomainID* part of the <GlobalKeyID>
333 element in a <SymkeyRequest> MAY be zero ("0"). When an SKMS manages symmetric keys for multiple
334 enterprises, the *DomainID* in the <GlobalKeyID> of a <SymkeyRequest> MUST be positive and non-zero. In
335 such a situation, the client application will request a symmetric key for the domain in which it is authorized to
336 request and receive keys.

337 The *DomainID* in the <GlobalKeyID> element of a <SymkeyResponse> MUST always be positive and non-
338 zero. It will typically contain the PEN of the domain to which the symmetric key belongs.

339 The *ServerID* part of the <GlobalKeyID> element MUST be a positive integer in the range of 0 (zero) to
340 18446744073709551615 (20-byte ASCII decimal).

341 The *ServerID* part of the <GlobalKeyID> element of a <SymkeyRequest> MUST always be zero ("0").

342 The *ServerID* part of the <GlobalKeyID> element of a <SymkeyResponse> MUST always be positive and non-
343 zero. It will typically contain the unique server identifier of the SKS server where the symmetric key was
344 generated.

345 The *KeyID* part of the <GlobalKeyID> element MUST be a positive integer in the range of 0 (zero) to
346 18446744073709551615 (20-byte ASCII decimal).

347 The *KeyID* part of the <GlobalKeyID> element of a <SymkeyRequest> MUST always be zero ("0").

348 The **KeyID** part of the <GlobalKeyID> element of a <SymkeyResponse> MUST always be positive and non-
349 zero. It will typically contain the unique key identifier of the symmetric key within the SKS server where the key
350 was generated.

351 **Example 1 – A <GlobalKeyID> value for a new symmetric key from an SKMS that serves a single domain:**

```
352 <ekmi:GlobalKeyID>0-0-0</ekmi:GlobalKeyID>
```

353 **Example 2 – A <GlobalKeyID> value for a new symmetric key for the domain with the PEN 10514, from an**
354 **SKMS that serves multiple domains:**

```
355 <ekmi:GlobalKeyID>10514-0-0</ekmi:GlobalKeyID>
```

356 **Example 3 – A <GlobalKeyID> value for the 16,777,215th symmetric key generated on 2nd SKS server for**
357 **an enterprise with the PEN 10514, in either a <SymkeyRequest> or a <SymkeyResponse>:**

```
358 <ekmi:GlobalKeyID>10514-2-16777215</ekmi:GlobalKeyID>
```

359 **Example 4 – The maximum <GlobalKeyID> value possible (a 62-byte ASCII decimal), in a**
360 **<SymkeyRequest> or <SymkeyResponse>:**

```
361 <ekmi:GlobalKeyID>  
362 18446744073709551615-18446744073709551615-18446744073709551615  
363 </ekmi:GlobalKeyID>
```

364 2.3 Element <KeyClasses> and <KeyClass>

365 The <KeyClasses> element of type **KeyClassesType**, when specified, identifies at least one <KeyClass>
366 element, but may specify an unbounded (unlimited) number of <KeyClass> elements within the <KeyClasses>
367 set.

368 **Schema Definition:**

```
369 <xsd:complexType name="KeyClassesType">  
370 <xsd:sequence>  
371 <xsd:element  
372 name="KeyClass"  
373 type="tns:KeyClassType"  
374 minOccurs="1"  
375 maxOccurs="unbounded"/>  
376 </xsd:sequence>  
377 </xsd:complexType>  
  
378  
379 <xsd:simpleType name="KeyClassType">  
380 <xsd:restriction base="xsd:string">  
381 <xsd:maxLength value="255"/>  
382 </xsd:restriction>  
383 </xsd:simpleType>
```

384 Client applications may request one or more symmetric keys conforming to one or more key classes required by
385 the application. If the client application is authorized to receive keys conforming to such key classes, the SKS
386 server will generate and supply them.

387
388 The <KeyClasses> element is useful only when requesting new symmetric keys, i.e. symmetric encryption keys
389 that have previously NOT been used for encrypting data. There is little reason for a client application to specify
390 the <KeyClasses> element when requesting an existing (escrowed) symmetric key, since the SKS server will
391 return the requested key to authorized clients with whatever key class is associated with the key regardless of
392 what key class is specified in the request. The key class will have been associated with the symmetric key at the
393 time of its generation and cannot be changed once associated with a key.
394

395 When more than one <GlobalKeyID> is specified in the request, there MAY be only one <KeyClass> element
396 within the <KeyClasses> set. When a key class is not specified in a request, it implies a request for symmetric
397 key(s) of a default key class configured at the SKS server. The default key class for a site is site-specific.
398

399 When the client requires more than one symmetric key, and each key needs to be of a different key class, there
400 MUST be only one <GlobalKeyID> element followed by as many <KeyClass> elements inside the
401 <KeyClasses> set as needed by the client application. (Example 5 in this section).
402

403 When a client requires many symmetric keys – say five keys – and two or more keys belong to the same key
404 class, the client MUST send multiple requests to the SKS server. One request will contain multiple
405 <GlobalKeyID> elements with one <KeyClass> element in the <KeyClasses> set, and the other request will
406 contain one <GlobalKeyID> element and multiple <KeyClass> elements within the <KeyClasses> set.
407 (Examples 4 and 5 in this section).

408 **Example 1 – A symmetric key request of a default key class (when no KeyClass is specified):**

```
409 <ekmi:SymkeyRequest xmlns:ekmi="http://docs.oasis-open.org/ekmi/2008/01">  
410 <ekmi:GlobalKeyID>10514-0-0</ekmi:GlobalKeyID>  
411 </ekmi:SymkeyRequest>
```

412 **Example 2 – A request for multiple new symmetric keys, each of a default key class:**

```
413 <ekmi:SymkeyRequest xmlns:ekmi="http://docs.oasis-open.org/ekmi/2008/01">  
414 <ekmi:GlobalKeyID>10514-0-0</ekmi:GlobalKeyID>  
415 <ekmi:GlobalKeyID>10514-0-0</ekmi:GlobalKeyID>  
416 </ekmi:SymkeyRequest>
```

417 **Example 3 – A request for a new symmetric key of a specific key class:**

```
418 <ekmi:SymkeyRequest xmlns:ekmi="http://docs.oasis-open.org/ekmi/2008/01">  
419 <ekmi:GlobalKeyID>10514-0-0</ekmi:GlobalKeyID>  
420 <ekmi:KeyClasses>  
421 <ekmi:KeyClass>256-Bit-Class</ekmi:KeyClass>  
422 </ekmi:KeyClasses>  
423 </ekmi:SymkeyRequest>
```

424 **Example 4 – A request for two new symmetric keys of the same key class for each symmetric key. Note**
425 **that if the FIN-FX key class was the default key class, a request as shown in Example 2 of this section**
426 **would result in the same response:**

```
427 <ekmi:SymkeyRequest xmlns:ekmi="http://docs.oasis-open.org/ekmi/2008/01">  
428 <ekmi:GlobalKeyID>10514-0-0</ekmi:GlobalKeyID>  
429 <ekmi:GlobalKeyID>10514-0-0</ekmi:GlobalKeyID>  
430 <ekmi:KeyClasses>  
431 <ekmi:KeyClass>FIN-FX</ekmi:KeyClass>  
432 </ekmi:KeyClasses>  
433 </ekmi:SymkeyRequest>
```

434 **Example 5 – A request for a four new symmetric keys of different key classes:**

```
435 <ekmi:SymkeyRequest xmlns:ekmi="http://docs.oasis-open.org/ekmi/2008/01">  
436 <ekmi:GlobalKeyID>10514-0-0</ekmi:GlobalKeyID>  
437 <ekmi:KeyClasses>  
438 <ekmi:KeyClass>EHR-DEF</ekmi:KeyClass>  
439 <ekmi:KeyClass>EHR-HOS</ekmi:KeyClass>  
440 <ekmi:KeyClass>EHR-INS</ekmi:KeyClass>  
441 <ekmi:KeyClass>EHR-PAT</ekmi:KeyClass>  
442 </ekmi:KeyClasses>  
443 </ekmi:SymkeyRequest>
```

444 2.4 Element <SymkeyResponse>

445 The <SymkeyResponse> element is one of two results returned by an **SKS** server upon being sent a valid and
446 authorized <SymkeyRequest> by a client application. The other result is a <SymkeyError> which will be
447 discussed in the next section.

448 While <SymkeyResponse> is a top-level element within this specification, it **MUST** be enclosed within a **SOAP**
449 **Body** element of a **SOAP Envelope** to conform to the security requirements of this specification. The **SOAP**
450 **Header** of the **SOAP Envelope** **MUST** enclose a **Security** element conforming to [WSS] with a **ValueType**
451 attribute containing the value <http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-x509-token-profile-1.0#X509v3>. The **Security** element must conform to all other requirements of the specified security profile
452 in [WSS] to form a well-formed, secure message.
453

454 Schema Definition:

```
455 <xsd:element name="SymkeyResponse">  
456 <xsd:complexType>  
457 <xsd:sequence>  
458 <xsd:element  
459 name="Symkey"  
460 type="tns:SymkeyType"  
461 minOccurs="0"  
462 maxOccurs="unbounded"/>  
463 <xsd:element  
464 name="SymkeyError"  
465 type="tns:SymkeyErrorType"  
466 minOccurs="0"  
467 maxOccurs="unbounded"/>  
468 </xsd:sequence>  
469 </xsd:complexType>  
470 </xsd:element>
```

471 The <SymkeyResponse> element consists of a sequence of two types of child elements - <Symkey> or
472 <SymkeyError> . The <SymkeyResponse> element **MAY** consist of either type of element or both types of
473 elements. When both elements are contained in a <SymkeyResponse>, all <Symkey> elements **MUST** precede
474 the first <SymkeyError> element.

475 1. <Symkey> [Optional]

476 This element of type **SymkeyType**, is returned by the **SKS** server in response to a successful
477 processing of a <SymkeyRequest>. There **MAY** be more than one <Symkey> element in the
478 <SymkeyResponse> if the client application made a request for multiple symmetric keys.
479

480 The <Symkey> element and the **SymkeyType** are specified in Section 2.5.
481

482 2. <SymkeyError> [Optional]

483 This element of type **SymkeyErrorType**, contains a response to a failed attempt in processing a
484 request for one or more symmetric keys. There **MAY** be more than one <SymkeyError> element in
485 the <SymkeyResponse> if the client application made a request for multiple symmetric keys and the
486 request resulted in multiple errors.
487

488 The <SymkeyError> element is specified in Section 2.6.
489

490 Some high-level examples of the <SymkeyResponse> element are as follows:

491 Example 1 – A response with a single symmetric key:

```
492 <ekmi:SymkeyResponse  
493 xmlns:ekmi="http://docs.oasis-open.org/ekmi/2008/01">
```

```
494     <ekmi:Symkey>.....</ekmi:Symkey>
495 </ekmi:SymkeyResponse>
```

496 **Example 2 – A response with three symmetric keys:**

```
497 <ekmi:SymkeyResponse
498   xmlns:ekmi="http://docs.oasis-open.org/ekmi/2008/01">
499   <ekmi:Symkey>.....</ekmi:Symkey>
500   <ekmi:Symkey>.....</ekmi:Symkey>
501   <ekmi:Symkey>.....</ekmi:Symkey>
502 </ekmi:SymkeyResponse>
```

503 **Example 3 – A response with an error:**

```
504 <ekmi:SymkeyResponse
505   xmlns:ekmi="http://docs.oasis-open.org/ekmi/2008/01">
506   <ekmi:SymkeyError>.....</ekmi:SymkeyError>
507 </ekmi:SymkeyResponse>
```

508 **Example 4 – A response with multiple errors:**

```
509 <ekmi:SymkeyResponse
510   xmlns:ekmi="http://docs.oasis-open.org/ekmi/2008/01">
511   <ekmi:SymkeyError>.....</ekmi:SymkeyError>
512   <ekmi:SymkeyError>.....</ekmi:SymkeyError>
513 </ekmi:SymkeyResponse>
```

514 **Example 5 – A response with one symmetric key and one error:**

```
515 <ekmi:SymkeyResponse
516   xmlns:ekmi="http://docs.oasis-open.org/ekmi/2008/01">
517   <ekmi:Symkey>.....</ekmi:Symkey>
518   <ekmi:SymkeyError>.....</ekmi:SymkeyError>
519 </ekmi:SymkeyResponse>
```

520 **Example 6 – A response with multiple symmetric keys and multiple error:**

```
521 <ekmi:SymkeyResponse
522   xmlns:ekmi="http://docs.oasis-open.org/ekmi/2008/01">
523   <ekmi:Symkey>.....</ekmi:Symkey>
524   <ekmi:Symkey>.....</ekmi:Symkey>
525   <ekmi:Symkey>.....</ekmi:Symkey>
526   <ekmi:SymkeyError>.....</ekmi:SymkeyError>
527   <ekmi:SymkeyError>.....</ekmi:SymkeyError>
528 </ekmi:SymkeyResponse>
```

529 **2.5 Element <Symkey>**

530 The <Symkey> element is the *raison d'être* of the SKSML protocol. The element of type **SymkeyType**, contains
531 the symmetric key returned by the SKS server, in response to a successful processing of a <SymkeyRequest>
532 from a client application.

533 **Schema Definition:**

```
534 <xsd:complexType name="SymkeyType">
535   <xsd:sequence>
536     <xsd:element name="GlobalKeyID" type="ekmi:GlobalKeyIDType"/>
537     <xsd:element name="KeyUsePolicy" type="ekmi:KeyUsePolicyType"/>
538     <xsd:element name="EncryptionMethod" type="xenc:EncryptionMethodType"/>
539     <xsd:element ref="xenc:CipherData"/>
540   </xsd:sequence>
541 </xsd:complexType>
```

542 When a request for a symmetric key is successful, there MUST be at least one <Symkey> element in a
543 <SymkeyResponse> element. There MAY be more than one <Symkey> element in the response if the client
544 application made a request for multiple symmetric keys and the SKS server processed the request successfully.

545 In the event of an error in processing the request, there SHALL be no <Symkey> element in the response; there
546 SHALL be a <SymkeyError> element, instead. The <SymkeyError> element is specified in Section 2.6.

547 The <Symkey> element consists of a sequence of the following child elements:

548 1. <GlobalKeyID> [Required]

549

550 This element of type **GlobalKeyIDType** identifies the unique identifier of the symmetric key within an
551 SKMS. There SHALL be only one <GlobalKeyID> within a <Symkey> element.

552

553 The **GlobalKeyIDType** is specified in Section 2.2.

554 2. <KeyUsePolicy> [Required]

555

556 This element of type **KeyUsePolicyType**, defines how the symmetric key in this <Symkey> element may
557 be used by applications. There SHALL be only one <KeyUsePolicy> element within a <Symkey>
558 element.

559

560 The <KeyUsePolicy> element is specified in Section 2.7.

561 3. <EncryptionMethod> [Required]

562

563 This element of type **EncryptionMethodType** from [XMLEncryption] describes how the symmetric key
564 in this <Symkey> element is encrypted for transport between the SKS Server and the client application.

565

566 The <EncryptionMethod> MUST specify one of the following two transport algorithms in the
567 **Algorithm** attribute of the element:

568

569 - http://www.w3.org/2001/04/xmlenc#rsa-1_5

570 - <http://www.w3.org/2001/04/xmlenc#rsa-oaep-mgf1p>

571 4. <CipherData> [Required]

572

573 This element of **CipherDataType** from [XMLEncryption] contains the encrypted symmetric-key. As
574 specified in [XMLEncryption], the content of this element is Base-64 encoded and is of the XML
575 Schema **base64Binary** type.

576 Some high-level examples of the <Symkey> element are as follows. Details about the <KeyUsePolicy>
577 element have been elided for brevity:

578 **Example 1 – A response with a symmetric key:**

```
579 <ekmi:SymkeyResponse
580   xmlns:ekmi="http://docs.oasis-open.org/ekmi/2008/01">
581   <ekmi:Symkey>
582     <ekmi:GlobalKeyID>10514-1-235</ekmi:GlobalKeyID>
583     <ekmi:KeyUsePolicy>....</ekmi:KeyUsePolicy>
584     <ekmi:EncryptionMethod
585       Algorithm="http://www.w3.org/2001/04/xmlenc#rsa-1_5"/>
586     <xenc:CipherData>
587       <xenc:CipherValue>
588         E9zWB/y93hVSzeTLiDcQoDxmLnXTux+SffMNwCJmt1dIqzQHBnpdQ8
589         1g6DKdkCFjJmHqhywCx9sfYjv9h5FDqUiQXG0ca8EU871zBoXBjDxj
590         fg1pU8tGFbpWZcd/ATpJD/UJow/qimxi8+huUYJMtaGH=
591       </xenc:CipherValue>
592     </xenc:CipherData>
593   </ekmi:Symkey>
594 </ekmi:SymkeyResponse>
```

595 **Example 2 – A response with multiple symmetric keys:**

```
596 <ekmi:SymkeyResponse
597   xmlns:ekmi="http://docs.oasis-open.org/ekmi/2008/01">
598   <ekmi:Symkey>
599     <ekmi:GlobalKeyID>10514-1-235</ekmi:GlobalKeyID>
600     <ekmi:KeyUsePolicy>.....</ekmi:KeyUsePolicy>
601     <ekmi:EncryptionMethod
602       Algorithm="http://www.w3.org/2001/04/xmlenc#rsa-oaep-mgf1p"/>
603     <xenc:CipherData>
604       <xenc:CipherValue>
605         E9zWB/y93hVSzeTLiDcQoDxmLNxTux+SffMNwCJmt1dIqzQHBnpdQ8
606         1g6DKdkCFjJMhQhywCx9sfYjv9h5FDqUiQXG0ca8EU871zBoXBjDxj
607         fg1pU8tGFbpWZcd/ATpJD/UJow/qimxi8+huUYJMtaGH=
608       </xenc:CipherValue>
609     </xenc:CipherData>
610   </ekmi:Symkey>
611   <ekmi:Symkey>
612     <ekmi:GlobalKeyID>10514-1-236</ekmi:GlobalKeyID>
613     <ekmi:KeyUsePolicy>.....</ekmi:KeyUsePolicy>
614     <ekmi:EncryptionMethod
615       Algorithm="http://www.w3.org/2001/04/xmlenc#rsa-oaep-mgf1p"/>
616     <xenc:CipherData>
617       <xenc:CipherValue>
618         Qbg65cy93hVSzeTLiDcQoDxmLNxTux+SffMNwCJmt1dIqzQHBnpdQ8
619         7k6DKdkCFjJMhQhywCx9sfYjv9h5FDqUiQXG0ca8EU871zBoXBjDxj
620         uyecU8tGFbpWZcd/ATpJD/UJow/qimxi8+huUYJMtaGH=
621       </xenc:CipherValue>
622     </xenc:CipherData>
623   </ekmi:Symkey>
624 </ekmi:SymkeyResponse>
```

625 **2.6 Element <SymkeyError>**

626 The <SymkeyError> element of type *SymkeyErrorType*, contains the error returned by the SKS server, in
627 response to a failure in processing of a <SymkeyRequest> from a client application.

628 **Schema Definition:**

```
629 <xsd:complexType name="SymkeyErrorType">
630   <xsd:sequence>
631     <xsd:element name="RequestedGlobalKeyID" type="ekmi:GlobalKeyIDType"/>
632     <xsd:element
633       name="RequestedKeyClass"
634       type="ekmi:KeyClassType"
635       minOccurs="0"/>
636     <xsd:element name="ErrorCode">
637       <xsd:simpleType>
638         <xsd:restriction base="xsd:string">
639           <xsd:maxLength value="255"/>
640         </xsd:restriction>
641       </xsd:simpleType>
642     </xsd:element>
643     <xsd:element name="ErrorMessage">
644       <xsd:simpleType>
645         <xsd:restriction base="xsd:string">
646           <xsd:maxLength value="1024"/>
647         </xsd:restriction>
648       </xsd:simpleType>
649     </xsd:element>
```

650 </xsd:sequence>
651 </xsd:complexType>

652 When a request for a symmetric key fails despite successfully being processed by the SOAP layer, there MUST
653 be at least one <SymkeyError> element in a <SymkeyResponse> element. When a <SymkeyRequest> fails
654 at the SOAP layer, the response SHALL consist of a **SOAPFault**.

655 There MAY be more than one <SymkeyError> element in the response if the client application made a request
656 for multiple symmetric keys and the **SKS** server failed in processing the request for more than one symmetric
657 key.

658 The <SymkeyError> element consists of a sequence of the following child elements:

659 1. <RequestedGlobalKeyID> [Required]

660
661 This element of type **GlobalKeyIDType** identifies the unique identifier of the symmetric key requested by
662 the client application. There SHALL be only one <RequestedGlobalKeyID> within a
663 <SymkeyError> element.

664
665 The **GlobalKeyIDType** is specified in Section 2.2.

666 2. <RequestedKeyClass> [Optional]

667
668 This element of type **KeyClassType** identifies the key-class of the symmetric key requested by the
669 client application. If the <RequestedKeyClass> element is not embedded in the <SymkeyError>
670 element, this implies that the requested symmetric key was for the default key-class of the SKMS.

671
672 The **KeyClassType** is specified in Section 2.3.

673 3. <ErrorCode> [Required]

674
675 This element of type **String** identifies a mnemonic code identifying the error the **SKS** Server
676 experienced in processing the client's symmetric key request.

677
678 The <ErrorCode> element SHALL return one of the codes identified in Appendix D of this specification.

679 4. <ErrorMessage> [Required]

680
681 This element of type **String** identifies a localized message describing the error the **SKS** Server
682 experienced in processing the client's symmetric key request.

683
684 The <ErrorMessage> element SHALL return the appropriate localized version of the message
685 corresponding to the <ErrorCode> element from Appendix D of this specification.

686 Some high-level examples of the <SymkeyError> element are as follows.

687 **Example 1 – An error within a response:**

```
688 <ekmi:SymkeyResponse  
689     xmlns:ekmi="http://docs.oasis-open.org/ekmi/2008/01">  
690     <ekmi:SymkeyError>  
691         <ekmi:RequestedGlobalKeyID>10514-2-22</ekmi:RequestedGlobalKeyID>  
692         <ekmi:ErrorCode>SKS-100004</ekmi:ErrorCode>  
693         <ekmi:ErrorMessage>Unauthorized request for key</ekmi:ErrorMessage>  
694     </ekmi:SymkeyError>  
695 </ekmi:SymkeyResponse>
```

696 **Example 2 – Multiple errors within a response:**

```
697 <ekmi:SymkeyResponse  
698     xmlns:ekmi="http://docs.oasis-open.org/ekmi/2008/01">  
699     <ekmi:SymkeyError>
```



```

700         <ekmi:RequestedGlobalKeyID>10514-1-0</ekmi:RequestedGlobalKeyID>
701         <ekmi:ErrorCode>SKS-100001</ekmi:ErrorCode>
702         <ekmi:ErrorMessage>Invalid GlobalKeyID</ekmi:ErrorMessage>
703     </ekmi:SymkeyError>
704     <ekmi:SymkeyError>
705         <ekmi:RequestedGlobalKeyID>10514-2-22</ekmi:RequestedGlobalKeyID>
706         <ekmi:ErrorCode>SKS-100004</ekmi:ErrorCode>
707         <ekmi:ErrorMessage>Unauthorized request for key</ekmi:ErrorMessage>
708     </ekmi:SymkeyError>
709 </ekmi:SymkeyResponse>

```

710 2.7 Element <KeyUsePolicy>

711 The <KeyUsePolicy> element defines rules that conforming implementations of the **SKCL** MUST adhere to
712 when using the symmetric key sent by the **SKS** Server. It is an integral part of the <Symkey> element .

713 Schema Definition:

```

714     <xsd:complexType name="KeyUsePolicyType" mixed="true">
715         <xsd:sequence>
716             <xsd:element name="KeyUsePolicyID" type="tns:TwoPartIDType"/>
717             <xsd:element name="PolicyName">
718                 <xsd:simpleType>
719                     <xsd:restriction base="xsd:string">
720                         <xsd:maxLength value="255"/>
721                     </xsd:restriction>
722                 </xsd:simpleType>
723             </xsd:element>
724             <xsd:element name="KeyClass" type="tns:KeyClassType"/>
725             <xsd:element name="KeyAlgorithm" type="tns:EncryptionAlgorithmType"/>
726             <xsd:element name="KeySize" type="tns:KeySizeType"/>
727             <xsd:element name="Status" type="tns:StatusType"/>
728             <xsd:element name="Permissions" type="tns:PermissionsType"/>
729         </xsd:sequence>
730     </xsd:complexType>

```

731 The <KeyUsePolicy> element is of the **KeyUsePolicyType** and consists of the following child elements:

- 732 1. <KeyUsePolicyID> [Required]
733
734 The <KeyUsePolicyID> element, of type **TwoPartIDType** , identifies the unique policy object within
735 the SKMS. There SHALL be only one <KeyUsePolicyID> element within a <KeyUsePolicy>
736 element.
737
738 The **TwoPartIDType** is specified in Section 2.8.
- 739 2. <PolicyName> [Required]
740
741 The <PolicyName> element, of type XSD **String**, with a maximum length of 255 characters,
742 identifies a unique name given to this <KeyUsePolicy>. There SHALL be only one <PolicyName>
743 element within a <KeyUsePolicy> element.
- 744 3. <KeyClass> [Required]
745
746 The <KeyClass> element ,of type **KeyClassType** , identifies a key-class assigned to this
747 <KeyUsePolicy>. There SHALL be only one <KeyUsePolicyID> element within a
748 <KeyUsePolicy> element.
749
750 The **KeyClassType** is specified in Section 2.3.

- 751 4. <KeyAlgorithm> [Required]
752
753 The <KeyAlgorithm> element , of type **EncryptionAlgorithmType** , identifies encryption algorithm to
754 be used by applications when using this symmetric key. There SHALL be only one <KeyAlgorithm>
755 element within a <KeyUsePolicy> element.
756
757 The <KeyAlgorithm> element is specified in Section 2.9.
- 758 5. <KeySize> [Required]
759
760 The <KeySize> element , of type **KeySizeType**, defines the size of the symmetric key, in bits (binary
761 digits). There SHALL be only one <KeySize> element within a <KeyUsePolicy> element.
762
763 Note: It is possible to determine the size of a symmetric key in an **SKCL** implementation without having
764 to send the size in the response. So, why include it? It is our belief that while network bandwidth and
765 compute performance of devices are increasing steadily, encryption is desired in many small and
766 portable devices. Consequently, it will speed up applications in cryptographic processing if they do not
767 have to determine the size of each key they use. While “protocol purity” demands that implementation
768 issues do not show up in protocol design, we believe it is justified in this case.
769
770 The **KeySizeType** is specified in Section 2.10.
- 771 6. <Status> [Required]
772
773 The <Status> element, of type **StatusType**, identifies the current status of the symmetric key. There
774 SHALL be only one <Status> element within a <KeyUsePolicy> element.
775
776 The **StatusType** is specified in Section 2.11.
- 777 7. <Permissions> [Required]
778
779 The <Permissions> element, of type **PermissionsType**, defines what is permissible to client
780 applications with the symmetric key this element is associated with. It is the responsibility of the
781 conforming **SKCL** implementation to enforce these rules.
782
783 An important distinction of this element – unlike most access control rules – is that the absence of sub-
784 elements in the <Permissions> element implies that all permissions are allowed. The presence of
785 sub-elements in this element provide rules to the **SKCL** about what actions are permitted.
786
787 There SHALL be only one <Permissions> element in a <KeyUsePolicy> element.
788
789 The **PermissionsType** is specified in Section 2.12.

790 Some examples of the <KeyUsePolicy> element are as follows.

791 **Example 1 – A <KeyUsePolicy> with some permission restrictions:**

```

792 <ekmi:KeyUsePolicy>
793   <ekmi:KeyUsePolicyID>10514-4</ekmi:KeyUsePolicyID>
794   <ekmi:PolicyName>DES-EDE KeyUsePolicy</ekmi:PolicyName>
795   <ekmi:KeyClass>HR-Class</ekmi:KeyClass>
796   <ekmi:KeyAlgorithm>
797     http://www.w3.org/2001/04/xmlenc#tripleDES-cbc
798   </ekmi:KeyAlgorithm>
799   <ekmi:KeySize>192</ekmi:KeySize>
800   <ekmi:Status>Active</ekmi:Status>
801   <ekmi:Permissions>
802     <ekmi:PermittedApplications ekmi:any="false">
803       <ekmi:PermittedApplication>
804         <ekmi:ApplicationID>10514-23</ekmi:ApplicationID>
805         <ekmi:ApplicationName>Payroll Application</ekmi:ApplicationName>

```

```

806         <ekmi:Version>1.0</ekmi:Version>
807         <ekmi:DigestAlgorithm>
808             http://www.w3.org/2000/09/xmlsig#sha1
809         </ekmi:DigestAlgorithm>
810         <ekmi:DigestValue>
811             229ea73a5e76eabd183663d332b283948a9202a1
812         </ekmi:DigestValue>
813     </ekmi:PermittedApplication>
814 </ekmi:PermittedApplications>
815 <ekmi:PermittedDates ekmi:any="false">
816     <ekmi:PermittedDate>
817         <ekmi:StartDate>2008-01-01</ekmi:StartDate>
818         <ekmi:EndDate>2008-12-31</ekmi:EndDate>
819     </ekmi:PermittedDate>
820 </ekmi:PermittedDates>
821 <ekmi:PermittedDays ekmi:any="true" xsi:nil="true"/>
822 <ekmi:PermittedDuration ekmi:any="true" xsi:nil="true"/>
823 <ekmi:PermittedLevels ekmi:any="true" xsi:nil="true"/>
824 <ekmi:PermittedLocations ekmi:any="true" xsi:nil="true"/>
825 <ekmi:PermittedNumberOfTransactions ekmi:any="true" xsi:nil="true"/>
826 <ekmi:PermittedTimes ekmi:any="false">
827     <ekmi:PermittedTime>
828         <ekmi:StartTime>07:00:00</ekmi:StartTime>
829         <ekmi:EndTime>19:00:00</ekmi:EndTime>
830     </ekmi:PermittedTime>
831 </ekmi:PermittedTimes>
832 <ekmi:PermittedUses ekmi:any="true" xsi:nil="true"/>
833 </ekmi:Permissions>
834 </ekmi:KeyUsePolicy>

```

835 **Example 2 – A <KeyUsePolicy> with no restrictions on the key:**

```

836 <ekmi:KeyUsePolicy>
837     <ekmi:KeyUsePolicyID>10514-2</ekmi:KeyUsePolicyID>
838     <ekmi:PolicyName>Laptop KeyUsePolicy</ekmi:PolicyName>
839     <ekmi:KeyClass>HR-Class</ekmi:KeyClass>
840     <ekmi:KeyAlgorithm>
841         http://www.w3.org/2001/04/xmlenc#aes256-cbc
842     </ekmi:KeyAlgorithm>
843     <ekmi:KeySize>256</ekmi:KeySize>
844     <ekmi:Status>Active</ekmi:Status>
845     <ekmi:Permissions>
846         <ekmi:PermittedApplications ekmi:any="true" xsi:nil="true"/>
847         <ekmi:PermittedDates ekmi:any="true" xsi:nil="true"/>
848         <ekmi:PermittedDays ekmi:any="true" xsi:nil="true"/>
849         <ekmi:PermittedDuration ekmi:any="true" xsi:nil="true"/>
850         <ekmi:PermittedLevels ekmi:any="true" xsi:nil="true"/>
851         <ekmi:PermittedLocations ekmi:any="true" xsi:nil="true"/>
852         <ekmi:PermittedNumberOfTransactions ekmi:any="true" xsi:nil="true"/>
853         <ekmi:PermittedTimes ekmi:any="true" xsi:nil="true"/>
854         <ekmi:PermittedUses ekmi:any="true" xsi:nil="true"/>
855     </ekmi:Permissions>
856 </ekmi:KeyUsePolicy>

```

857 **2.8 Type TwoPartIDType**

858 The *TwoPartIDType* is used to create identifiers for many elements within the SKSML. It is a simple
859 concatenation of two integers with a hyphen between them ("-") to create an XML Schema *String* type.

860 The *TwoPartIDType* has a minimum length of three (3) characters, and a maximum length of 41 characters.

861 **Schema Definition:**

```
862 <xsd:simpleType name="TwoPartIDType">
863   <xsd:restriction base="xsd:string">
864     <xsd:minLength value="3"/>
865     <xsd:maxLength value="41"/>
866     <xsd:pattern value="[1-9][0-9]{0,19}-[1-9][0-9]{0,19}"/>
867     <xsd:whiteSpace value="collapse"/>
868   </xsd:restriction>
869 </xsd:simpleType>
```

870 The *TwoPartIDType* is used in the <ApplicationID>, the <KeyCachePolicyID> and the <KeyUsePolicyID>
871 elements within the SKSML.

872 Some examples of the <KeyUsePolicy> element are as follows.

873 **Example 1 – A *TwoPartIDType* used to identify an ApplicationID:**

```
874 <ekmi:ApplicationID>10514-23</ekmi:ApplicationID>
```

875 **Example 2 – A *TwoPartIDType* used to identify a KeyUsePolicyID:**

```
876 <ekmi:KeyUsePolicyID>10514-4</ekmi:KeyUsePolicyID>
```

877 **Example 3 – A minimum-length *TwoPartIDType* :**

```
878 <ekmi:KeyCachePolicyID>5-4</ekmi:KeyCachePolicyID>
```

879 **Example 4 – A maximum-length *TwoPartIDType* :**

```
880 <ekmi:ApplicationID>
881   18446744073709551615-18446744073709551615
882 </ekmi:ApplicationID>
```

883 2.9 Element <KeyAlgorithm>

884 The element <KeyAlgorithm> , of type *EncryptionAlgorithmType*, is used to identify the cryptographic
885 algorithm to be used with the symmetric keys in the <SymkeyResponse>.

886 **Schema Definition:**

```
887 <xsd:simpleType name="EncryptionAlgorithmType">
888   <xsd:restriction base="xsd:anyURI">
889     <xsd:enumeration value="http://www.w3.org/2001/04/xmlenc#tripledes-cbc"/>
890     <xsd:enumeration value="http://www.w3.org/2001/04/xmlenc#aes128-cbc"/>
891     <xsd:enumeration value="http://www.w3.org/2001/04/xmlenc#aes192-cbc"/>
892     <xsd:enumeration value="http://www.w3.org/2001/04/xmlenc#aes256-cbc"/>
893   </xsd:restriction>
894 </xsd:simpleType>
```

895 The algorithms currently supported by this specification are the algorithms defined in [XMLEncryption]. As new
896 algorithms are added to [XMLEncryption], they will be added to the enumerated list in this element. Currently,
897 the following four algorithms are supported:

898 1. Triple Data Encryption Standard (3DES)

899

900 Within the context of this specification, and as specified in [XMLEncryption], the form of 3DES
901 supported within SKSML is a 192-bit key with a 64-bit Initialization Vector. Of the key bits, the first 64
902 are used in the first DES operation, the second 64 bits in the second (middle) DES operation, and the
903 third 64 bits in the third (last) DES operation. Each of these 64 bits of key contain 56 effective bits and
904 8 parity bits.

- 905 The algorithm standard is denoted by the URL: <http://www.w3.org/2001/04/xmlenc#tripleDES-cbc>.
- 906
- 907 2. Advanced Encryption Standard (AES) – 128-bit
- 908
- 909 Within the context of this specification, and as specified in [AES], this is a 128-bit symmetric key used in
- 910 the Cipher Block Chaining (CBC) mode.
- 911
- 912 The algorithm standard is denoted by the URL: <http://www.w3.org/2001/04/xmlenc#aes128-cbc>.
- 913 3. Advanced Encryption Standard (AES) – 192-bit
- 914
- 915 Within the context of this specification, and as specified in [AES], this is a 192-bit symmetric key used in
- 916 the Cipher Block Chaining (CBC) mode.
- 917
- 918 The algorithm standard is denoted by the URL: <http://www.w3.org/2001/04/xmlenc#aes192-cbc>.
- 919 4. Advanced Encryption Standard (AES) – 256-bit
- 920
- 921 Within the context of this specification, and as specified in [AES], this is a 256-bit symmetric key used in
- 922 the Cipher Block Chaining (CBC) mode.
- 923
- 924 The algorithm standard is denoted by the URL: <http://www.w3.org/2001/04/xmlenc#aes256-cbc>.

925 There SHALL be only one <KeyAlgorithm> element within a <KeyUsePolicy> element.

926 Some examples of the <KeyAlgorithm> element are as follows; other elements of the <KeyUsePolicy>

927 element are not displayed for brevity:

928 **Example 1 – An example using the Triple-DES key algorithm:**

```
929 <ekmi:KeyUsePolicy>
930   ...
931   <ekmi:KeyAlgorithm>
932     http://www.w3.org/2001/04/xmlenc#tripleDES-cbc
933   </ekmi:KeyAlgorithm>
934   ...
935 </ekmi:KeyUsePolicy>
```

936 **Example 2 – An example using the AES-128 key algorithm:**

```
937 <ekmi:KeyUsePolicy>
938   ...
939   <ekmi:KeyAlgorithm>
940     http://www.w3.org/2001/04/xmlenc#aes128-cbc
941   </ekmi:KeyAlgorithm>
942   ...
943 </ekmi:KeyUsePolicy>
```

944 **2.10 Element <KeySize>**

945 The element <KeySize>, of type *KeySizeType*, is used to identify the size of the symmetric key, in binary digits

946 (bits) in the <SymkeyResponse>.

947 **Schema Definition:**

```
948 <xsd:simpleType name="KeySizeType">
949   <xsd:restriction base="xsd:unsignedShort">
950     <xsd:totalDigits value="3"/>
951     <xsd:fractionDigits value="0"/>
952     <xsd:enumeration value="128"/>
```

```

953         <xsd:enumeration value="192"/>
954         <xsd:enumeration value="256"/>
955     </xsd:restriction>
956 </xsd:simpleType>

```

957 There SHALL be only one <KeySize> element within a <KeyUsePolicy> element.

958 Currently, the following three key-sizes are supported:

- 959 1. 128-bits when used with the *AES-192* algorithm
- 960 2. 192-bits when used with the *AES-192* or the *3DES* algorithms
- 961 3. 256-bits when used with the *AES-256* algorithm

962 Some examples of the <KeySize> element are as follows; other elements of the <KeyUsePolicy> element are
963 not displayed for brevity:

964 **Example 1 – An example using a 128-bit key size:**

```

965 <ekmi:KeyUsePolicy>
966     ...
967     <ekmi:KeySize>128</ekmi:KeySize>
968     ...
969 </ekmi:KeyUsePolicy>

```

970 **Example 2 – An example using a 192-bit key size:**

```

971 <ekmi:KeyUsePolicy>
972     ...
973     <ekmi:KeySize>192</ekmi:KeySize>
974     ...
975 </ekmi:KeyUsePolicy>

```

976 **Example 3 – An example using a 256-bit key size:**

```

977 <ekmi:KeyUsePolicy>
978     ...
979     <ekmi:KeySize>256</ekmi:KeySize>
980     ...
981 </ekmi:KeyUsePolicy>

```

982 2.11 Element <Status>

983 The element <Status>, of type *StatusType*, is used to identify the current status of an object . It is used in
984 almost every element within the SKMS.

985 **Schema Definition:**

```

986 <xsd:simpleType name="StatusType">
987     <xsd:restriction base="xsd:string">
988         <xsd:enumeration value="Active"/>
989         <xsd:enumeration value="Default"/>
990         <xsd:enumeration value="Inactive"/>
991         <xsd:enumeration value="Other"/>
992     </xsd:restriction>
993 </xsd:simpleType>

```

994 Where it exists within an element, there SHALL be only one <Status> element within the enclosing element.

995 The <Status> element can contain one of four *String* type values:

- 996 1. The **Active** value indicates that the element that makes up the document-root is currently active in the
997 SKMS and conforming **SKCL** implementations may use it within applications.
- 998 2. The **Default** value indicates that the element that makes up the document root is the default element in
999 the SKMS, is also active, and conforming **SKCL** implementations may use it within applications.
- 1000 3. The **Inactive** value indicates that the element that makes up the document root is not active in the
1001 SKMS, and conforming **SKCL** implementations may NOT use it within applications.
- 1002 4. The **Other** value indicates that the element that makes up the document root has a meaning that is
1003 application-specific. However, conforming **SKCL** implementations may NOT use it within applications.

1004 Some examples of the <Status> element are shown below; other parts of their enclosing elements are not
1005 shown for brevity:

1006 **Example 1 – An example with an Active status within a <KeyUsePolicy> element:**

```
1007 <ekmi:KeyUsePolicy>
1008   ...
1009   <ekmi:Status>Active</ekmi:Status>
1010   ...
1011 </ekmi:KeyUsePolicy>
```

1012 **Example 2 – An example with an Inactive status within a <KeyUsePolicy> element:**

```
1013 <ekmi:KeyUsePolicy>
1014   ...
1015   <ekmi:Status>Inactive</ekmi:Status>
1016   ...
1017 </ekmi:KeyUsePolicy>
```

1018 **Example 3 – An example with a Default status within a <KeyUsePolicy> element:**

```
1019 <ekmi:KeyUsePolicy>
1020   ...
1021   <ekmi:Status>Default</ekmi:Status>
1022   ...
1023 </ekmi:KeyUsePolicy>
```

1024 2.12 Element <Permissions>

1025 The <Permissions> element, of the type *PermissionsType* is at the heart of the <KeyUsePolicy> element. It
1026 provides guidance to conforming **SKCL** implementations on who may use the symmetric key, when they may use
1027 it, for what purposes, for how long and in which locations. For applications that conform to the Multi-Level
1028 Security (MLS) model, there is a provision for specifying which levels are permitted use of the key. There is also
1029 an element that allows for extending the <Permissions> element to accommodate rules that have not been
1030 envisioned in the current specification.

1031 There SHALL be only one <Permissions> element within a <KeyUsePolicy> element.

1032 **Schema Definition:**

```
1033 <xsd:complexType name="PermissionsType">
1034 <xsd:sequence>
1035 <xsd:element
1036     name="PermittedApplications"
1037     type="tns:PermittedApplicationsType"
1038     minOccurs="1"
1039     nillable="true"/>
1040 <xsd:element
1041     name="PermittedDates"
1042     type="tns:PermittedDatesType"
```

```

1043         minOccurs="1"
1044         nillable="true"/>
1045     <xsd:element
1046         name="PermittedDays"
1047         type="tns:PermittedDaysType"
1048         minOccurs="1"
1049         nillable="true"/>
1050     <xsd:element
1051         name="PermittedDuration"
1052         type="tns:PermittedDurationType"
1053         minOccurs="1"
1054         nillable="true"/>
1055     <xsd:element
1056         name="PermittedLevels"
1057         type="tns:PermittedLevelsType"
1058         minOccurs="1"
1059         nillable="true"/>
1060     <xsd:element
1061         name="PermittedLocations"
1062         type="tns:PermittedLocationsType"
1063         minOccurs="1"
1064         nillable="true"/>
1065     <xsd:element
1066         name="PermittedNumberOfTransactions"
1067         type="tns:PermittedNumberOfTransactionsType"
1068         minOccurs="1"
1069         nillable="true"/>
1070     <xsd:element
1071         name="PermittedTimes"
1072         type="tns:PermittedTimesType"
1073         minOccurs="1"
1074         nillable="true"/>
1075     <xsd:element
1076         name="PermittedUses"
1077         type="tns:PermittedUsesType"
1078         minOccurs="1"
1079         nillable="true"/>
1080     <xsd:element
1081         name="Other"
1082         type="xsd:anyType"
1083         minOccurs="0"/>
1084 </xsd:sequence>
1085 </xsd:complexType>

```

1086
1087 The <Permissions> element consists of the following sub-elements:

1088 1. A required <PermittedApplications> element which identifies applications that are permitted use of
1089 the symmetric key in question. While the <PermittedApplications> element is required, it may be
1090 empty (NULL). The absence of sub-elements in the <PermittedApplications> element implies that
1091 all applications are permitted to use the key. Identifying a specific application restricts the use of the key
1092 to only the identified applications.

1093 The <PermittedApplications> element is specified in Section 2.13.
1094

1095 2. A required <PermittedDates> element which identifies the date ranges during which applications are
1096 permitted use of the symmetric key in question. While the <PermittedDates> element is required, it
1097 may be empty (NULL). The absence of sub-elements in the <PermittedDates> element implies that
1098 applications are permitted to use the key on any date. Identifying specific date ranges restricts the use
1099 of the key to only the duration between the identified dates.

1100
1101 The <PermittedDates> element is specified in Section 2.12.

1102 3. A required <PermittedDays> element which identifies the days of week during which applications are
1103 permitted use of the symmetric key in question. While the <PermittedDays> element is required, it
1104 may be empty (NULL). The absence of sub-elements in the <PermittedDays> element implies that
1105 applications are permitted to use the key on any day of the week. Identifying specific days restricts the
1106 use of the key to only the identified days.
1107
1108 The <PermittedDays> element is specified in Section 2.15.

1109 4. A required <PermittedDuration> element which identifies the duration (in seconds) in which
1110 applications are permitted use of the symmetric key in question *once the SKCL starts using the*
1111 *symmetric key*. While the <PermittedDuration> element is required, it may be empty (NULL). The
1112 absence of any content – the duration time - in the <PermittedDuration> element implies that
1113 applications are permitted to use the key for any duration after it has been used. Identifying a non-zero,
1114 positive duration value restricts the use of the key to only the period after the start of the use of the key.
1115
1116 A distinction between <PermittedDates> and <PermittedDuration> is that the former has fixed
1117 start and end-dates for the use of the key, whereas the latter has a fixed end-date-and-time after the
1118 key has begun to be used without a fixed start-date-and-time. Thus, an application with a
1119 <PermittedDuration> can begin the use of a symmetric key at any time, but must stop its use at the
1120 end of the <PermittedDuration> once it has begun. With <PermittedDates>, an application can
1121 continue using the symmetric key until the fixed date-and-time have been reached.
1122
1123 The <PermittedDuration> element is specified in Section 2.x16

1124 5. Within a Multi-Level Security (MLS) system, the required <PermittedLevels> element identifies the
1125 security levels at which applications are permitted use of the symmetric key in question. While the
1126 <PermittedLevels> element is required, it may be empty (NULL). The absence of sub-elements in
1127 the <PermittedLevels> element implies that applications are permitted to use the key at any level of
1128 security. Identifying specific MLS level(s) restricts the use of the key to only the identified security
1129 level(s).
1130
1131 The <PermittedLevels> element is specified in Section 2.x17

1132 6. A required <PermittedLocations> element which identifies physical geographic locations where
1133 applications are permitted use of the symmetric key in question. While the <PermittedLocations>
1134 element is required, it may be empty (NULL). The absence of sub-elements in the
1135 <PermittedLocations> element implies that applications are permitted to use the key at any physical
1136 location. Identifying specific locations restricts the use of the key to only the identified locations.
1137
1138 The <PermittedLocations> element is specified in Section 2.18.

1139 7. A required <PermittedNumberOfTransactions> element which identifies the number of encryption
1140 transactions that applications are permitted, with the use of the symmetric key in question. While the
1141 <PermittedNumberOfTransactions> element is required, it may be empty (NULL). The absence of
1142 content – the number of transactions – in the <PermittedNumberOfTransactions> element implies
1143 that applications are permitted to use the key for as many encryption transactions as necessary.
1144 Identifying a specific, non-zero, positive number of transactions in this element restricts the use of the
1145 key to only the limit identified in the element.
1146
1147 The <PermittedNumberOfTransactions> element is specified in Section 2.19.

1148 8. A required <PermittedTimes> element which identifies the times of day during which applications are
1149 permitted the use of the symmetric key in question. While the <PermittedTimes> element is required,
1150 it may be empty (NULL). The absence of sub-elements in the <PermittedTimes> element implies that
1151 applications are permitted to use the key at any time of day. Identifying specific times restricts the use
1152 of the key to only the duration of the identified times.
1153
1154 The <PermittedTimes> element is specified in Section 2.20.

1155 9. A required <PermittedUses> element which identifies application-uses that applications are permitted
1156 with the symmetric key in question. While the <PermittedUses> element is required, it may be empty
1157 (NULL). The absence of sub-elements in the <PermittedUses> element implies that applications are
1158 permitted to use the key for any purpose. Identifying specific uses restricts the use of the key to only
1159 the identified uses.

1160 The <PermittedUses> element is specified in Section 2.21.

1162 10. The optional <Other> element allows implementers to specify permissions that cannot be addressed
1163 with the above-mentioned categories, for restricting the use of the symmetric key in question.

1164 While the <Other> element provides flexibility for implementations, the disadvantage of the element is
1165 that it may render a specific implementation incompatible with other SKMS implementations that use the
1166 SKSML standard.

1167 **It is strongly recommended that implementers avoid the use of the <Other> element unless they**
1168 **definitely do not expect to inter-operate with other SKCL implementations. If there is a strong**
1169 **need for capability that does not exist within the current specification of the <Permissions>**
1170 **element, implementers are encouraged to contact the OASIS EKMI TC with their requirements.**
1171

1172 When all sub-elements of the <Permissions> element are empty, there are no restrictions on the use of the
1173 symmetric key other than that the application calling on the SKCL be authorized to access the key in question.
1174 However, when there are elements defined within the sub-elements of the <Permissions> element, conforming
1175 SKCL implementations must comply with all the permission elements, evaluating the most restrictive permissions
1176 first and in decreasing order of restriction, before allowing the use of the key.
1177

1178 For example, if a <Permissions> element specifies that a key may be used on Weekdays, between the hours
1179 of 0900 and 1700 Hours, then a request for a symmetric key on a Saturday at 1105 would deny use of the key in
1180 question, since it violates the more restrictive permission of being allowed for use only on weekdays.

1181 **It should be noted that it is the primary responsibility of a conforming SKCL to enforce the**
1182 **<Permission> elements' rules. The SKS server will generate the key – or return an existing key - when**
1183 **an authorized client with appropriate access requests it. However, it is up to the SKCL implementation to**
1184 **comply with the rules in the <Permissions> element.**

1185 In another example, if a <Permissions> element specifies a <PermittedDuration> of 600 seconds from the
1186 start of use of the key, and there is also present a <PermittedNumberOfTransactions> element with a value
1187 of 10 (encryption transactions), conforming SKCL implementations must evaluate both permissions before each
1188 transaction and determine if they are both within the specified thresholds before using the key. If the 600
1189 seconds expire before the 10 encryption transactions have been completed, or if the 10 encryption transactions
1190 are completed before 600 seconds have expired, conforming SKCL implementations MUST not use the key in
1191 question anymore.

1192 Some examples of the <Permissions> element are as shown below; the enclosing <KeyUsePolicy> element,
1193 <Symkey> element and <SymkeyResponse> elements are not displayed for brevity:

1194 **Example 1 – A <Permissions> element that permits a single application the use of the symmetric key in**
1195 **question, between January 01, 2008 and December 31, 2008 and between the hours of 0700 and 1900.**
1196 **There are, however, no restrictions on what days of the week the key may be used, the locations where it**
1197 **may be used, at which MLS level it may be used (if it applies), the number of data files/transactions that**
1198 **may be encrypted with the key or the uses of the key within that application:**

```
1199 <ekmi:Permissions>  
1200   <ekmi:PermittedApplications ekmi:any="false">  
1201     <ekmi:PermittedApplication>  
1202       <ekmi:ApplicationID>10514-23</ekmi:ApplicationID>  
1203       <ekmi:ApplicationName>Payroll Application</ekmi:ApplicationName>  
1204       <ekmi:Version>1.0</ekmi:Version>  
1205       <ekmi:DigestAlgorithm>  
1206         http://www.w3.org/2000/09/xmldsig#sha1  
1207       </ekmi:DigestAlgorithm>
```

```

1208         <ekmi:DigestValue>
1209             229ea73a5e76eabd183663d332b283948a9202a1
1210         </ekmi:DigestValue>
1211     </ekmi:PermittedApplication>
1212 </ekmi:PermittedApplications>
1213 <ekmi:PermittedDates ekmi:any="false">
1214     <ekmi:PermittedDate>
1215         <ekmi:StartDate>2008-01-01</ekmi:StartDate>
1216         <ekmi:EndDate>2008-12-31</ekmi:EndDate>
1217     </ekmi:PermittedDate>
1218 </ekmi:PermittedDates>
1219 <ekmi:PermittedDays ekmi:any="true" xsi:nil="true"/>
1220 <ekmi:PermittedDuration ekmi:any="true" xsi:nil="true"/>
1221 <ekmi:PermittedLevels ekmi:any="true" xsi:nil="true"/>
1222 <ekmi:PermittedLocations ekmi:any="true" xsi:nil="true"/>
1223 <ekmi:PermittedNumberOfTransactions ekmi:any="true" xsi:nil="true"/>
1224 <ekmi:PermittedTimes ekmi:any="false">
1225     <ekmi:PermittedTime>
1226         <ekmi:StartTime>07:00:00</ekmi:StartTime>
1227         <ekmi:EndTime>19:00:00</ekmi:EndTime>
1228     </ekmi:PermittedTime>
1229 </ekmi:PermittedTimes>
1230 <ekmi:PermittedUses ekmi:any="true" xsi:nil="true"/>
1231 </ekmi:Permissions>

```

1232 **Example 2 – A <Permissions> element that permits two specific applications the use of the symmetric**
1233 **key in question, between January 01, 2009 and January 31, 2009.**

```

1234 <ekmi:Permissions>
1235     <ekmi:PermittedApplications ekmi:any="false">
1236         <ekmi:PermittedApplication>
1237             <ekmi:ApplicationID>10514-24</ekmi:ApplicationID>
1238             <ekmi:ApplicationName>
1239                 Employee Tax Reporting Application
1240             </ekmi:ApplicationName>
1241             <ekmi:Version>3.3</ekmi:Version>
1242             <ekmi:DigestAlgorithm>
1243                 http://www.w3.org/2000/09/xmldsig#sha1
1244             </ekmi:DigestAlgorithm>
1245             <ekmi:DigestValue>
1246                 af96d65a7a2415239c8eb8be1347f704322957a4
1247             </ekmi:DigestValue>
1248         </ekmi:PermittedApplication>
1249     <ekmi:PermittedApplication>
1250         <ekmi:ApplicationID>10514-25</ekmi:ApplicationID>
1251         <ekmi:ApplicationName>
1252             IRS Tax Reporting Application
1253         </ekmi:ApplicationName>
1254         <ekmi:Version>2.1</ekmi:Version>
1255         <ekmi:DigestAlgorithm>
1256             http://www.w3.org/2000/09/xmldsig#sha1
1257         </ekmi:DigestAlgorithm>
1258         <ekmi:DigestValue>
1259             a4f5925185ffe12c1a91ea3de90fc086b34b34b2
1260         </ekmi:DigestValue>
1261     </ekmi:PermittedApplication>
1262 </ekmi:PermittedApplications>
1263 <ekmi:PermittedDates ekmi:any="false">
1264     <ekmi:PermittedDate>
1265         <ekmi:StartDate>2009-01-01</ekmi:StartDate>

```

```

1266         <ekmi:EndDate>2009-12-31</ekmi:EndDate>
1267     </ekmi:PermittedDate>
1268 </ekmi:PermittedDates>
1269 <ekmi:PermittedDays ekmi:any="true" xsi:nil="true"/>
1270 <ekmi:PermittedDuration ekmi:any="true" xsi:nil="true"/>
1271 <ekmi:PermittedLevels ekmi:any="true" xsi:nil="true"/>
1272 <ekmi:PermittedLocations ekmi:any="true" xsi:nil="true"/>
1273 <ekmi:PermittedNumberOfTransactions ekmi:any="true" xsi:nil="true"/>
1274 <ekmi:PermittedTimes ekmi:any="true" xsi:nil="true"/>
1275 <ekmi:PermittedUses ekmi:any="true" xsi:nil="true"/>
1276 </ekmi:Permissions>

```

1277 **Example 3 – A <Permissions> element that permits all applications the use of the symmetric key in**
1278 **question, for 100 transactions for encrypting credit card numbers.**

```

1279 <ekmi:Permissions>
1280     <ekmi:PermittedApplications ekmi:any="true" xsi:nil="true"/>
1281     <ekmi:PermittedDates ekmi:any="true" xsi:nil="true"/>
1282     <ekmi:PermittedDays ekmi:any="true" xsi:nil="true"/>
1283     <ekmi:PermittedDuration ekmi:any="true" xsi:nil="true"/>
1284     <ekmi:PermittedLevels ekmi:any="true" xsi:nil="true"/>
1285     <ekmi:PermittedLocations ekmi:any="true" xsi:nil="true"/>
1286     <ekmi:PermittedNumberOfTransactions ekmi:any="false">
1287         100
1288     </ekmi:PermittedNumberOfTransactions>
1289     <ekmi:PermittedTimes ekmi:any="true" xsi:nil="true"/>
1290     <ekmi:PermittedUses ekmi:any="false">
1291         <ekmi:PermittedUse>CCN</ekmi:PermittedUse>
1292     </ekmi:PermittedUses>
1293 </ekmi:Permissions>

```

1294 **Example 4 – A <Permissions> element that permits all applications the use of the symmetric key in**
1295 **question, for 600 seconds once the SKCL starts using the key.**

```

1296 <ekmi:Permissions>
1297     <ekmi:PermittedApplications ekmi:any="true" xsi:nil="true"/>
1298     <ekmi:PermittedDates ekmi:any="true" xsi:nil="true"/>
1299     <ekmi:PermittedDays ekmi:any="true" xsi:nil="true"/>
1300     <ekmi:PermittedDuration ekmi:any="false">600</ekmi:PermittedDuration>
1301     <ekmi:PermittedLevels ekmi:any="true" xsi:nil="true"/>
1302     <ekmi:PermittedLocations ekmi:any="true" xsi:nil="true"/>
1303     <ekmi:PermittedNumberOfTransactions ekmi:any="true" xsi:nil="true"/>
1304     <ekmi:PermittedTimes ekmi:any="true" xsi:nil="true"/>
1305     <ekmi:PermittedUses ekmi:any="true" xsi:nil="true"/>
1306 </ekmi:Permissions>

```

1307 **Example 5 – A <Permissions> element that permits a specific application the use of the symmetric key**
1308 **in question, at specific geographic locations only on weekdays between the hours of 0800 and 1700, and**
1309 **only when the application is operating at the Secret security level within an MLS system.**

```

1310 <ekmi:Permissions>
1311     <ekmi:PermittedApplications ekmi:any="true" xsi:nil="true"/>
1312     <ekmi:PermittedDates ekmi:any="true" xsi:nil="true"/>
1313     <ekmi:PermittedDays ekmi:any="false">
1314         <ekmi:PermittedDay>Weekday</ekmi:PermittedDay>
1315     </ekmi:PermittedDays>
1316     <ekmi:PermittedDuration ekmi:any="true" xsi:nil="true"/>
1317     <ekmi:PermittedLevels ekmi:any="false">
1318         <ekmi:PermittedLevel>Secret</ekmi:PermittedLevel>
1319     </ekmi:PermittedLevels>
1320     <ekmi:PermittedLocations ekmi:any="false">

```

```

1321     <ekmi:PermittedLocation>
1322         <ekmi:LocationName>Facility A51</ekmi:LocationName>
1323         <ekmi:Latitude>37.385562</ekmi:Latitude>
1324         <ekmi:Longitude>-121.993387</ekmi:Longitude>
1325     </ekmi:PermittedLocation>
1326     <ekmi:PermittedLocation>
1327         <ekmi:LocationName>Facility DC-VA01</ekmi:LocationName>
1328         <ekmi:Latitude>88.485362</ekmi:Latitude>
1329         <ekmi:Longitude>-21.453648</ekmi:Longitude>
1330     </ekmi:PermittedLocation>
1331 </ekmi:PermittedLocations>
1332 <ekmi:PermittedNumberOfTransactions ekmi:any="true" xsi:nil="true"/>
1333 <ekmi:PermittedTimes ekmi:any="false">
1334     <ekmi:PermittedTime>
1335         <ekmi:StartTime>08:00:00</ekmi:StartTime>
1336         <ekmi:EndTime>17:00:00</ekmi:EndTime>
1337     </ekmi:PermittedTime>
1338 </ekmi:PermittedTimes>
1339 <ekmi:PermittedUses ekmi:any="true" xsi:nil="true"/>
1340 </ekmi:Permissions>

```

1341 2.13 Element <PermittedApplications> and <PermittedApplication>

1342 The element <PermittedApplications>, of type *PermittedApplicationsType* and its only child-element
1343 <PermittedApplication> of type *ApplicationsType* are used to define the list of applications that are
1344 permitted to use a symmetric key within a specific <Symkey> element.

1345 Schema Definition:

```

1346 <xsd:complexType name="PermittedApplicationsType">
1347     <xsd:sequence>
1348         <xsd:element
1349             name="PermittedApplication"
1350             type="tns:ApplicationsType"
1351             minOccurs="0"
1352             maxOccurs="unbounded"/>
1353     </xsd:sequence>
1354     <xsd:attribute ref="tns:any" use="required"/>
1355 </xsd:complexType>

```

1356 Schema Definition:

```

1357 <xsd:complexType name="ApplicationsType">
1358     <xsd:sequence>
1359         <xsd:element name="ApplicationID" type="tns:TwoPartIDType"/>
1360         <xsd:element name="ApplicationName">
1361             <xsd:simpleType>
1362                 <xsd:restriction base="xsd:string">
1363                     <xsd:maxLength value="256"/>
1364                     <xsd:whiteSpace value="preserve"/>
1365                 </xsd:restriction>
1366             </xsd:simpleType>
1367         </xsd:element>
1368         <xsd:element name="Version" minOccurs="0">
1369             <xsd:simpleType>
1370                 <xsd:restriction base="xsd:string">
1371                     <xsd:maxLength value="32"/>
1372                     <xsd:whiteSpace value="preserve"/>
1373                 </xsd:restriction>
1374             </xsd:simpleType>

```

```

1375         </xsd:simpleType>
1376     </xsd:element>
1377     <xsd:group ref="tns:MessageDigestGroup" minOccurs="0"/>
1378     <xsd:element name="Other" type="xsd:anyType" minOccurs="0"/>
1379 </xsd:sequence>
1380 </xsd:complexType>

```

1381 **Schema Definition:**

```

1382
1383 <xsd:group name="MessageDigestGroup">
1384   <xsd:sequence>
1385     <xsd:element name="DigestAlgorithm">
1386       <xsd:simpleType>
1387         <xsd:restriction base="xsd:anyURI">
1388           <xsd:enumeration value="http://www.w3.org/2000/09/xmlsig#sha1"/>
1389           <xsd:enumeration value="http://www.w3.org/2001/04/xmlenc#sha256"/>
1390           <xsd:enumeration value="http://www.w3.org/2001/04/xmlenc#sha512"/>
1391         </xsd:restriction>
1392       </xsd:simpleType>
1393     </xsd:element>
1394     <xsd:element name="DigestValue">
1395       <xsd:simpleType>
1396         <xsd:restriction base="xsd:base64Binary">
1397           <xsd:maxLength value="1024"/>
1398         </xsd:restriction>
1399       </xsd:simpleType>
1400     </xsd:element>
1401   </xsd:sequence>
1402 </xsd:group>

```

1403 **Schema Definition:**

```

1404   <xsd:attribute name="any">
1405     <xsd:simpleType>
1406       <xsd:restriction base="xsd:string">
1407         <xsd:enumeration value="false"/>
1408         <xsd:enumeration value="true"/>
1409       </xsd:restriction>
1410     </xsd:simpleType>
1411   </xsd:attribute>

```

1412 There SHALL be only one <PermittedApplications> element within the <Permissions> element.
 1413 However, there MAY be an unbounded (unlimited) number of <PermittedApplication> elements within a
 1414 <PermittedApplications> element.

1415 The <PermittedApplications> element SHALL have one attribute named "any", that will have a "false" or
 1416 "true" value, based on the following:

- 1417 • When the <PermittedApplications> element is null (i.e. it does not have a single
 1418 <PermittedApplication> sub-element in it), the value of the "any" attribute SHALL be set to "true"
 1419 AND the XML Schema Instance (XSI) "nil" attribute SHALL be set to "true".
- 1420 • When the <PermittedApplications> element is not-null (i.e. it has at least one
 1421 <PermittedApplication> sub-element in it), the value of the "any" attribute SHALL be set to "false"
 1422 AND the XML Schema Instance (XSI) "nil" attribute SHALL NOT be present.

1423 A null <PermittedApplications> element specifies that ALL applications are permitted use of the symmetric
 1424 key, subject to complying with all other permission clauses in the <KeyUsePolicy> element.

1425 The <PermittedApplication> sub-element of type **ApplicationsType**, provides details of the application
 1426 which is permitted use of the symmetric key in question. The <PermittedApplication> element consists of
 1427 the following sub-elements:

- 1427 1. The <ApplicationID> element identifies the unique identifier assigned to this application within the
 1428 SKMS. It is a **TwoPartIDType** as specified in Section 2.8. There SHALL be only one
 1429 <ApplicationID> element within a <PermittedApplication> element.
- 1430 2. The <ApplicationName> element identifies the name assigned to this application within the SKMS. It
 1431 is an XSD **String** with a maximum length of 256 characters. There SHALL be only one
 1432 <ApplicationName> element within a <PermittedApplication> element.
- 1433 3. An optional <Version> element identifying the version number of this application within the SKMS. It
 1434 is an XSD **String** with a maximum length of 32 characters. There MAY be only one <Version>
 1435 element within a <PermittedApplication> element.
- 1436 4. The <MessageDigestGroup> group which identifies the message digest value of the application's
 1437 binary image, along with the message digest algorithm used to calculate the digest value. The
 1438 <MessageDigestGroup> consists of the following elements:
- 1439 a) The <DigestAlgorithm> element of the XSD type **anyURI**, which supports one of the
 1440 following three digest algorithms:
- 1441 i. <http://www.w3.org/2000/09/xmldsig#sha1>
- 1442 ii. <http://www.w3.org/2001/04/xmlenc#sha256>
- 1443 iii. <http://www.w3.org/2001/04/xmlenc#sha512>
- 1444 b) The <DigestValue> element of the XSD type **base64Binary**.
- 1445 There SHALL be only one <MessageDigestGroup> group within a <PermittedApplication>
 1446 element.
- 1447 5. An optional <Other> element that provides implementers the ability to carry other information about
 1448 the application that may be relevant to their SKMS. Implementers are cautioned that the use of the
 1449 <Other> element may not be supported by other SKCL implementations, and may break
 1450 interoperability between two SKMS implementations. Should there be a strong need for additional
 1451 features in the <PermittedApplication> element, implementers are encouraged to contact the
 1452 OASIS EKMI TC with their requirements.

1453 NOTE: The SKSML specification does not specify how an SKCL implementation will determine the message
 1454 digest of an application that needs to use the symmetric key in question. It is left to the implementers of the
 1455 SKCL to determine the message digest of the calling application using the specified algorithm, and verify that
 1456 the digest values match before the SKCL uses the symmetric key on behalf of the application.

1457 Some examples of the <PermittedApplications> element are shown below; other parts of their enclosing
 1458 elements are not shown for brevity:

1459 **Example 1 – An example of a <PermittedApplications> element with two child**
 1460 **<PermittedApplication> elements with specific version numbers and message digest values:**

```

1461 <ekmi:PermittedApplications ekmi:any="false">
1462   <ekmi:PermittedApplication>
1463     <ekmi:ApplicationID>10514-24</ekmi:ApplicationID>
1464     <ekmi:ApplicationName>Employee Tax Reporting</ekmi:ApplicationName>
1465     <ekmi:Version>3.3</ekmi:Version>
1466     <ekmi:DigestAlgorithm>
1467       http://www.w3.org/2000/09/xmldsig#sha1
1468     </ekmi:DigestAlgorithm>
1469     <ekmi:DigestValue>G4bsdfKkt4cziEqFFu0oBTM81efU=</ekmi:DigestValue>
1470   </ekmi:PermittedApplication>
1471   <ekmi:PermittedApplication>
1472     <ekmi:ApplicationID>10514-25</ekmi:ApplicationID>
1473     <ekmi:ApplicationName>IRS Tax Reporting Application</ekmi:ApplicationName>
1474     <ekmi:Version>2.1</ekmi:Version>
1475     <ekmi:DigestAlgorithm>

```

```

1476         http://www.w3.org/2001/04/xmlenc#sha256
1477     </ekmi:DigestAlgorithm>
1478     <ekmi:DigestValue>
1479         ab7b85c9410d48c54fc7939c391be4028e7305085191c56e7b3740f2cbdbbc79
1480     </ekmi:DigestValue>
1481 </ekmi:PermittedApplication>
1482 </ekmi:PermittedApplications>

```

1483 **Example 2 – An example of a <PermittedApplications> element with one child**
1484 **<PermittedApplication> element that applies to all versions of the application; the message digest**
1485 **value and algorithm are not used in this example:**

```

1486 <ekmi:PermittedApplications ekmi:any="false">
1487     <ekmi:PermittedApplication>
1488         <ekmi:ApplicationID>10514-14</ekmi:ApplicationID>
1489         <ekmi:ApplicationName>E-Commerce Payment</ekmi:ApplicationName>
1490     </ekmi:PermittedApplication>
1491 </ekmi:PermittedApplications>

```

1492 **Example 3 – An example of a null <PermittedApplications> element specifying that ALL applications**
1493 **are permitted the use of the symmetric key:**

```

1494 <ekmi:PermittedApplications ekmi:any="true" xsi:nil="true"/>

```

1495 2.14 Element <PermittedDates> and <PermittedDate>

1496 The element <PermittedDates>, of type *PermittedDatesType* and its only child-element <PermittedDate>, which is an anonymous XSD *ComplexType*, are used to define ranges of dates between which applications are permitted to use the symmetric key within a specific <Symkey> element.

1499 Schema Definition:

```

1500 <xsd:complexType name="PermittedDatesType">
1501     <xsd:sequence>
1502         <xsd:element name="PermittedDate" minOccurs="0" maxOccurs="unbounded">
1503             <xsd:complexType>
1504                 <xsd:sequence>
1505                     <xsd:element name="StartDate">
1506                         <xsd:simpleType>
1507                             <xsd:restriction base="xsd:date">
1508                                 <xsd:pattern value="\p{Nd}{4}-\p{Nd}{2}-\p{Nd}{2}"/>
1509                             </xsd:restriction>
1510                         </xsd:simpleType>
1511                     </xsd:element>
1512                     <xsd:element name="EndDate">
1513                         <xsd:simpleType>
1514                             <xsd:restriction base="xsd:date">
1515                                 <xsd:pattern value="\p{Nd}{4}-\p{Nd}{2}-\p{Nd}{2}"/>
1516                             </xsd:restriction>
1517                         </xsd:simpleType>
1518                     </xsd:element>
1519                 </xsd:sequence>
1520             </xsd:complexType>
1521         </xsd:element>
1522     </xsd:sequence>
1523     <xsd:attribute ref="tns:any" use="required"/>
1524 </xsd:complexType>

```

1525 There SHALL be only one <PermittedDates> element within the <Permissions> element. However, there
1526 MAY be an unbounded number of <PermittedDate> elements within a <PermittedDates> element.

1527 The <PermittedDates> element SHALL have one attribute named "any", that will have a "false" or "true" value,
1528 based on the following:

1529 • When the <PermittedDates> element is null (i.e. it does not have a single <PermittedDate> sub-
1530 element in it), the value of the "any" attribute SHALL be set to "true" AND the XML Schema Instance
1531 (XSI) "nil" attribute SHALL be set to "true".

1532 • When the <PermittedDates> element is not-null (i.e. it has at least one <PermittedDate> sub-
1533 element in it), the value of the "any" attribute SHALL be set to "false" AND the XML Schema Instance
1534 (XSI) "nil" attribute SHALL NOT be present.

1535 A null <PermittedDates> element specifies that applications are permitted use of the symmetric key on any
1536 calendar date of the year, subject to complying with all other permission clauses in the <Permissions>
1537 element.

1538 The <PermittedDate> sub-element identifies an individual set of dates between which application are
1539 permitted use of the symmetric key in question. The <PermittedDate> element consists of the following sub-
1540 elements:

1541 1. The <StartDate> element identifies the date from which applications MAY start using the symmetric
1542 key in question. It is an XSD **Date** type that MUST be specified in a specific pattern (see examples)
1543 where the first four digits specify the year, the second two digits specify the calendar month in the year,
1544 and the last two digits specify the calendar date of the month.

1545 There SHALL be only one <StartDate> element within a <PermittedDate> element.

1546 Conforming SKCL implementations SHALL NOT start using the symmetric before the onset of the
1547 <StartDate> on the client machine. Unless further constrained by the <PermittedTimes> element,
1548 the onset of the <StartDate> is specified to be the first second of the day – 00:00:01 Hours – on the
1549 client machine.
1550

1551 2. The <EndDate> element identifies the date until which applications may use the symmetric key in
1552 question. It is an XSD **Date** type that MUST be specified in a specific pattern (see examples) where the
1553 first four digits specify the year, the second two digits specify the calendar month in the year, and the
1554 last two digits specify the calendar date of the month.

1555 There SHALL be only one <EndDate> element within a <PermittedDate> element.

1556 Conforming SKCL implementations SHALL NOT use the symmetric after the end of the <EndDate> on
1557 the client machine. Unless further constrained by the <PermittedTimes> element, the end of the
1558 <EndDate> is specified to be the last second of the day – 23:59:59 Hours – on the client machine.
1559

1560 Examples of the <PermittedDates> element are shown below; other required parts of their enclosing elements
1561 are not shown for brevity:

1562 **Example 1 – An example of a <PermittedDates> element with a single <PermittedDate> element. The**
1563 **<StartDate> specifies January 01, 2009 while the <EndDate> specifies January 31, 2009:**

```
1564 <ekmi:PermittedDates ekmi:any="false">  
1565 <ekmi:PermittedDate>  
1566 <ekmi:StartDate>2009-01-01</ekmi:StartDate>  
1567 <ekmi:EndDate>2009-01-31</ekmi:EndDate>  
1568 </ekmi:PermittedDate>  
1569 </ekmi:PermittedDates>
```

1570 **Example 2 – An example of a <PermittedDates> element with two <PermittedDate> elements. For the**
1571 **first <PermittedDate> element, the <StartDate> element specifies July 01, 2008 while the <EndDate>**
1572 **element specifies July 03, 2008. For the second <PermittedDate> element, the <StartDate> element**
1573 **specifies July 07, 2008 while the <EndDate> element specifies July 12, 2008. This policy would restrict a**
1574 **symmetric key with this <PermittedDates> element so it cannot be used on the July 4th weekend of**
1575 **2008:**
1576
1577

```

1578     <ekmi:PermittedDates ekmi:any="false">
1579         <ekmi:PermittedDate>
1580             <ekmi:StartDate>2008-07-01</ekmi:StartDate>
1581             <ekmi:EndDate>2008-07-03</ekmi:EndDate>
1582         </ekmi:PermittedDate>
1583         <ekmi:PermittedDate>
1584             <ekmi:StartDate>2008-07-07</ekmi:StartDate>
1585             <ekmi:EndDate>2009-07-12</ekmi:EndDate>
1586         </ekmi:PermittedDate>
1587     </ekmi:PermittedDates>

```

1588 **Example 3 – An example of a null <PermittedDates> element, specifying that applications are permitted**
1589 **use of the symmetric key on any date:**

```

1590     <ekmi:PermittedDates ekmi:any="true" xsi:nil="true"/>

```

1591 2.15 Element <PermittedDays> and <PermittedDay>

1592 The element <PermittedDays> , of the type *PermittedDaysType* and its only child-element <PermittedDay>
1593 of the *PermittedDayType*, are used to define days of the week on which applications are permitted to use a
1594 symmetric key within a specific <Symkey> element.

1595 Schema Definition:

```

1596     <xsd:complexType name="PermittedDaysType">
1597         <xsd:sequence>
1598             <xsd:element
1599                 name="PermittedDay"
1600                 type="tns:PermittedDayType"
1601                 minOccurs="0"
1602                 maxOccurs="unbounded">
1603             </xsd:element>
1604         </xsd:sequence>
1605         <xsd:attribute ref="tns:any" use="required"/>
1606     </xsd:complexType>

```

1607 Schema Definition:

```

1608     <xsd:simpleType name="PermittedDayType">
1609         <xsd:restriction base="xsd:string">
1610             <xsd:enumeration value="Sunday"/>
1611             <xsd:enumeration value="Monday"/>
1612             <xsd:enumeration value="Tuesday"/>
1613             <xsd:enumeration value="Wednesday"/>
1614             <xsd:enumeration value="Thursday"/>
1615             <xsd:enumeration value="Friday"/>
1616             <xsd:enumeration value="Saturday"/>
1617             <xsd:enumeration value="Weekday"/>
1618             <xsd:enumeration value="Weekend"/>
1619         </xsd:restriction>
1620     </xsd:simpleType>

```

1621 There SHALL be only one <PermittedDays> element within the <Permissions> element. However, there
1622 MAY be an unbounded (unlimited) number of <PermittedDay> elements within a <PermittedDays> element.

1623 The <PermittedDays> element SHALL have one attribute named “any”, that will have a “false” or “true” value,
1624 based on the following:

- 1625 • When the <PermittedDays> element is null (i.e. it does not have a single <PermittedDay> sub-
1626 element in it), the value of the “any” attribute SHALL be set to “true” AND the XML Schema Instance
1627 (XSI) “nil” attribute SHALL be set to “true”.

- 1628 • When the <PermittedDays> element is not-null (i.e. it has at least one <PermittedDay> sub-
1629 element in it), the value of the "any" attribute SHALL be set to "false" AND the XML Schema Instance
1630 (XSI) "nil" attribute SHALL NOT be present.

1631 A null <PermittedDays> element specifies that applications are permitted use of the symmetric key on all days
1632 of the week, subject to complying with all other permission clauses in the <Permissions> element.

1633 The <PermittedDay> element, of the XSD *String* type, identifies individual days of the week from an
1634 enumerated list on which application are permitted to use the symmetric key in question.

1635 Examples of the <PermittedDays> element are shown below; other parts of their enclosing elements are not
1636 shown for brevity:

1637 **Example 1 – An example of a <PermittedDays> element with a single<PermittedDay> child element,**
1638 **specifying that the symmetric key may be used only on weekdays:**

```
1639 <ekmi:PermittedDays ekmi:any="false">  
1640 <ekmi:PermittedDay>Weekday</ekmi:PermittedDay>  
1641 </ekmi:PermittedDays>
```

1642 **Example 2 – An example of a <PermittedDays> element with three <PermittedDay> child elements,**
1643 **specifying that the symmetric key may be used only on Mondays, Wednesdays and Fridays:**

```
1644 <ekmi:PermittedDays ekmi:any="false">  
1645 <ekmi:PermittedDay>Monday</ekmi:PermittedDay>  
1646 <ekmi:PermittedDay>Wednesday</ekmi:PermittedDay>  
1647 <ekmi:PermittedDay>Friday</ekmi:PermittedDay>  
1648 </ekmi:PermittedDays>
```

1649 **Example 3 – An example of a null <PermittedDays> element, specifying that the symmetric key may be**
1650 **used on any day of the week:**

```
1651 <ekmi:PermittedDays ekmi:any="true" xsi:nil="true"/>
```

1652 **2.16 Element <PermittedDuration>**

1653 The element <PermittedDuration>, of the type *PermittedDurationType* is used to define the number of
1654 seconds, applications are permitted to use a symmetric key, once the SKCL has started using the symmetric key
1655 in question.

1656 **Schema Definition:**

```
1657 <xsd:complexType name="PermittedDurationType">  
1658 <xsd:simpleContent>  
1659 <xsd:extension base="tns:DurationType">  
1660 <xsd:attribute ref="tns:any" use="required"/>  
1661 </xsd:extension>  
1662 </xsd:simpleContent>  
1663 </xsd:complexType>
```

1664 **Schema Definition:**

```
1665 <xsd:simpleType name="DurationType">  
1666 <xsd:restriction base="xsd:positiveInteger">  
1667 <xsd:minInclusive value="1"/>  
1668 <xsd:maxInclusive value="18446744073709551615"/>  
1669 </xsd:restriction>  
1670 </xsd:simpleType>
```

1671 There SHALL be only one <PermittedDuration> element within the <Permissions> element.

1672 The <PermittedDuration> element SHALL have one attribute, named "any" that will have a "false" or "true"
1673 value, based on the following:

- 1674 • When the <PermittedDuration> element is null (i.e. it does not have any content in it), the value of
1675 the "any" attribute SHALL be set to "true" AND the XML Schema Instance (XSI) "nil" attribute SHALL be
1676 set to "true".
- 1677 • When the <PermittedDuration> element is not-null (i.e. it has content in it), the value of the "any"
1678 attribute SHALL be set to "false" AND the XML Schema Instance (XSI) "nil" attribute SHALL NOT be
1679 present.

1680 A null <PermittedDuration> element specifies that applications are permitted use of the symmetric key
1681 indefinitely, subject to complying with all other permission clauses in the <Permissions> element.

1682 The <PermittedDuration> element, of the XSD *positiveInteger* type, identifies the number of seconds for
1683 which the symmetric key in question may be used, ONCE the key has been used by conforming **SKCL**
1684 implementations for the first time. The values for <PermittedDuration> may range between 1 and
1685 18446744073709551615.

1686 As long as the symmetric has not been used by an **SKCL** on a client device (it might be cached for many
1687 days/weeks/months depending on the <KeyCachePolicy> in effect within the SKMS for that device) the
1688 effective lifetime of the symmetric key may be well past the number of seconds specified in
1689 <PermittedDuration> when calculated from the time of the key's generation time on the **SKS** server. It is the
1690 responsibility of the **SKCL** implementation, when presented with a <PermittedDuration> element in a
1691 <KeyUsePolicy> of a symmetric key, to keep track of the date/time when the symmetric key in question was
1692 first used on the client device, and how long the key will last after that.

1693 Examples of the <PermittedDuration> element are shown below; other parts of their enclosing elements are
1694 not shown for brevity:

1695 **Example 1 – An example of a <PermittedDuration> element specifying that the symmetric key may be
1696 used only for a single 24-hour period from the moment it is first used by an SKCL:**

```
1697 <ekmi:PermittedDuration ekmi:any="false">86400</ekmi:PermittedDuration>
```

1698 **Example 2 – An example of a <PermittedDuration> element specifying that the symmetric key may be
1699 used only for week from the moment it is first used by an SKCL:**

```
1700 <ekmi:PermittedDuration ekmi:any="false">604800</ekmi:PermittedDuration>
```

1701 **Example 3 – An example of a <PermittedDuration> element specifying that the symmetric key may be
1702 used only 5 minutes from the moment it is first used by an SKCL:**

```
1703 <ekmi:PermittedDuration ekmi:any="false">300</ekmi:PermittedDuration>
```

1704 **Example 4 – An example of a null <PermittedDuration> element specifying that the symmetric key may
1705 be used indefinitely by an SKCL:**

```
1706 <ekmi:PermittedDuration ekmi:any="true" xsi:nil="true"/>
```

1707 **2.17 Element <PermittedLevels> and <PermittedLevel>**

1708 The element <PermittedLevels>, of the type *LevelClassificationType*, is used to define the security level at
1709 which applications are permitted use of a symmetric key. This element is useful only to applications and systems
1710 that conform to the multi-level security (MLS) system as defined in the Bell-LaPadula model.

1711 **Schema Definition:**

```
1712 <xsd:complexType name="PermittedLevelsType">  
1713 <xsd:sequence>  
1714 <xsd:element  
1715 name="PermittedLevel"/>
```

```

1715         type="tns:LevelClassificationType"
1716         minOccurs="0"
1717         maxOccurs="unbounded">
1718     </xsd:element>
1719     <xsd:element name="Other" type="xsd:anyType" minOccurs="0"/>
1720 </xsd:sequence>
1721 <xsd:attribute ref="tns:any" use="required"/>
1722 </xsd:complexType>

```

1723 **Schema Definition:**

```

1724 <xsd:simpleType name="LevelClassificationType">
1725     <xsd:restriction base="xsd:string">
1726         <xsd:enumeration value="Unclassified"/>
1727         <xsd:enumeration value="Confidential"/>
1728         <xsd:enumeration value="Secret"/>
1729         <xsd:enumeration value="Top-Secret"/>
1730     </xsd:restriction>
1731 </xsd:simpleType>

```

1732 There SHALL be only one <PermittedLevels> element within the <Permissions> element. However, there
1733 MAY be an unbounded (unlimited) number of <PermittedLevel> elements within the <PermittedLevels>
1734 element. (Practically, it makes no sense to have more than the known levels; however, this specification leaves
1735 itself open to the possibility that other levels may be defined).

1736 The <PermittedLevels> element SHALL have one attribute named "any", that will have a "false" or "true"
1737 value, based on the following:

- 1738 • When the <PermittedLevels> element is null (i.e. it does not have a single <PermittedLevel> sub-
1739 element in it), the value of the "any" attribute SHALL be set to "true" AND the XML Schema Instance
1740 (XSI) "nil" attribute SHALL be set to "true".
- 1741 • When the <PermittedLevels> element is not-null (i.e. it has at least one <PermittedLevel> sub-
1742 element in it), the value of the "any" attribute SHALL be set to "false" AND the XML Schema Instance
1743 (XSI) "nil" attribute SHALL NOT be present.

1744 A null <PermittedLevels> element specifies that applications at ANY level are permitted use of the symmetric
1745 key, subject to complying with all other permission clauses in the <Permissions> element.

1746 The <PermittedLevel> sub-element, of the *LevelClassificationType*, identifies the precise MLS level at
1747 which the symmetric key in question may be used. The <PermittedLevel> SHALL contain one of the following
1748 four (4) enumerated values:

- 1749 1. Unclassified
- 1750 2. Confidential
- 1751 3. Secret
- 1752 4. Top-Secret

1753 Examples of the <PermittedLevels> element are shown below; other parts of their enclosing elements are
1754 not shown for brevity:

1755 **Example 1 – An example of a <PermittedLevels> element specifying that the symmetric key may be
1756 used only by applications at the Confidential level:**

```

1757 <ekmi:PermittedLevels ekmi:any="false">
1758     <ekmi:PermittedLevel>Confidential</ekmi:PermittedLevel>
1759 </ekmi:PermittedLevels>

```

1760 **Example 2 – An example of a <PermittedLevels> element specifying that the symmetric key may be
1761 used only by applications at the Secret or Top-Secret level:**

```

1762     <ekmi:PermittedLevels ekmi:any="false">
1763         <ekmi:PermittedLevel>Secret</ekmi:PermittedLevel>
1764         <ekmi:PermittedLevel>Top-Secret</ekmi:PermittedLevel>
1765     </ekmi:PermittedLevels>

```

1766 **Example 3 – An example of a null <PermittedLevels> element specifying that the symmetric key may**
1767 **be used at any level:**

```

1768     <ekmi:PermittedLevels ekmi:any="true" xsi:nil="true"/>

```

1769 2.18 Element <PermittedLocations> and <PermittedLocation>

1770 The element <PermittedLocations>, of the type *PermittedLocationsType*, is used to define the
1771 geographically physical locations where applications are permitted use of a symmetric key. This element is
1772 useful only to applications that have the ability to determine the Global Positioning System (GPS) location of the
1773 client device intending to use the symmetric key.

1774 Schema Definition:

```

1775     <xsd:complexType name="PermittedLocationsType">
1776         <xsd:sequence>
1777             <xsd:element name="PermittedLocation" minOccurs="1" maxOccurs="unbounded">
1778                 <xsd:complexType>
1779                     <xsd:sequence>
1780                         <xsd:element name="LocationName">
1781                             <xsd:simpleType>
1782                                 <xsd:restriction base="xsd:string">
1783                                     <xsd:maxLength value="256"/>
1784                                     <xsd:whiteSpace value="preserve"/>
1785                                 </xsd:restriction>
1786                             </xsd:simpleType>
1787                         </xsd:element>
1788                         <xsd:group
1789                             ref="tns:LocationCoordinateGroup"
1790                             minOccurs="0"
1791                             maxOccurs="unbounded"/>
1792                         <xsd:element name="Other" type="xsd:anyType" minOccurs="0"/>
1793                     </xsd:sequence>
1794                 </xsd:complexType>
1795             </xsd:element>
1796         </xsd:sequence>
1797     <xsd:attribute ref="tns:any" use="required"/>
1798 </xsd:complexType>

```

1799 Schema Definition:

```

1800     <xsd:group name="LocationCoordinateGroup">
1801         <xsd:sequence>
1802             <xsd:element name="Latitude">
1803                 <xsd:simpleType>
1804                     <xsd:restriction base="xsd:decimal">
1805                         <xsd:totalDigits value="10"/>
1806                         <xsd:fractionDigits value="7"/>
1807                     </xsd:restriction>
1808                 </xsd:simpleType>
1809             </xsd:element>
1810             <xsd:element name="Longitude">
1811                 <xsd:simpleType>
1812                     <xsd:restriction base="xsd:decimal">
1813                         <xsd:totalDigits value="10"/>

```

```

1814         <xsd:fractionDigits value="7"/>
1815         </xsd:restriction>
1816     </xsd:simpleType>
1817 </xsd:element>
1818 </xsd:sequence>
1819 </xsd:group>

```

1820 There SHALL be only one <PermittedLocations> element within the <Permissions> element. However,
 1821 there MAY be an unbounded (unlimited) number of <PermittedLocation> sub-elements within the
 1822 <PermittedLocations> element.

1823 The <PermittedLocations> element SHALL have one attribute named “any”, that will have a “false” or “true”
 1824 value, based on the following:

- 1825 • When the <PermittedLocations> element is null (i.e. it does not have a single
 1826 <PermittedLocation> sub-element in it), the value of the “any” attribute SHALL be set to “true” AND
 1827 the XML Schema Instance (XSI) “nil” attribute SHALL be set to “true”.
- 1828 • When the <PermittedLocations> element is not-null (i.e. it has at least one
 1829 <PermittedLocation> sub-element in it), the value of the “any” attribute SHALL be set to “false”
 1830 AND the XML Schema Instance (XSI) “nil” attribute SHALL NOT be present.

1831 A null <PermittedLocations> element specifies that applications are permitted use of the symmetric key at
 1832 ANY physical location, subject to complying with all other permission clauses in the <Permissions> element.

1833 The <PermittedLocation> element, of the *PermittedLocationType*, identifies the precise geographical
 1834 location where the symmetric key in question may be used. The <PermittedLocation> SHALL contain the
 1835 following elements:

- 1836 1. The <LocationName> element identifies a human-readable name of the physical location. It is an
 1837 XSD *String* type element, with a maximum length of 256 characters.

1838 There SHALL be only one <LocationName> element within a <PermittedLocation> element.
 1839

- 1840 2. An optional **LocationCoordinateGroup** which, when present, SHALL contain the following two
 1841 elements:

- 1842 a) The <Latitude> element of XSD *Decimal* type, that identifies the horizontal coordinate
 1843 location of the client device on the Earth, measured in *degrees* and expressed as a decimal
 1844 with the *minutes* and *seconds* part of the measurement expressed as a single fraction.

1845 When used, there SHALL be only one <Latitude> element within the
 1846 <PermittedLocation> element.
 1847

- 1848 b) The <Longitude> element of XSD *Decimal* type, that identifies the vertical coordinate location
 1849 of the client device on the Earth, measured in *degrees* and expressed as a decimal with the
 1850 *minutes* and *seconds* part of the measurement expressed as a single fraction.

1851 When used, there SHALL be only one <Longitude> element within the
 1852 <PermittedLocation> element.
 1853

1854 Some examples of the <PermittedLocations> element are shown below; other parts of their enclosing
 1855 elements are not shown for brevity:

1856 **Example 1 – An example of a <PermittedLocations> element specifying that the symmetric key may be
 1857 used only by applications at a single named location:**

```

1858 <ekmi:PermittedLocations ekmi:any="false">
1859   <ekmi:PermittedLocation>
1860     <ekmi:LocationName>StrongAuth Server Room</ekmi:LocationName>
1861   </ekmi:PermittedLocation>
1862 </ekmi:PermittedLocations>

```

1863 **Example 2 – An example of a <PermittedLocations> element specifying that the symmetric key may be**
1864 **used only by applications at a single location at the given GPS coordinates:**

```
1865     <ekmi:PermittedLocations ekmi:any="false">
1866         <ekmi:PermittedLocation>
1867             <ekmi:LocationName>StrongAuth Server Room</ekmi:LocationName>
1868             <ekmi:Latitude>37.385653 </ekmi:Latitude>
1869             <ekmi:Longitude>-121.993192 </ekmi:Longitude>
1870         </ekmi:PermittedLocation>
1871     </ekmi:PermittedLocations>
```

1872 **Example 3 – An example of a <PermittedLocations> element specifying that the symmetric key may be**
1873 **used only by applications at multiple locations:**

```
1874     <ekmi:PermittedLocations ekmi:any="false">
1875         <ekmi:PermittedLocation>
1876             <ekmi:LocationName>Humongous Headquarters</ekmi:LocationName>
1877         </ekmi:PermittedLocation>
1878         <ekmi:PermittedLocation>
1879             <ekmi:LocationName> Humongous Primary Data Center</ekmi:LocationName>
1880             <ekmi:Latitude>37.385653 </ekmi:Latitude>
1881             <ekmi:Longitude>-121.993192 </ekmi:Longitude>
1882         </ekmi:PermittedLocation>
1883         <ekmi:PermittedLocation>
1884             <ekmi:LocationName>Humongous DR Data Center</ekmi:LocationName>
1885             <ekmi:Latitude>68.845901 </ekmi:Latitude>
1886             <ekmi:Longitude>11.393385 </ekmi:Longitude>
1887         </ekmi:PermittedLocation>
1888     </ekmi:PermittedLocations>
```

1889 **Example 4 – An example of a null <PermittedLocations> element specifying that the symmetric key**
1890 **may be used at any location on the planet:**

```
1891     <ekmi:PermittedLocations ekmi:any="true" xsi:nil="true"/>
```

1892 **2.19 Element <PermittedNumberOfTransactions>**

1893 The element <PermittedNumberOfTransactions>, of type *PermittedNumberOfTransactionsType* is used
1894 to define the number of *encryption* transactions that applications are permitted with a symmetric key within a
1895 specific <Symkey> element, once the **SKCL** has started using the symmetric key in question. It does not limit
1896 the number of *decryption* transactions with the same symmetric key.

1897 **Schema Definition:**

```
1898     <xsd:complexType name="PermittedNumberOfTransactionsType">
1899         <xsd:simpleContent>
1900             <xsd:extension base="tns:NumberOfTransactionsType">
1901                 <xsd:attribute ref="tns:any" use="required"/>
1902             </xsd:extension>
1903         </xsd:simpleContent>
1904     </xsd:complexType>
```

1905 **Schema Definition:**

```
1906     <xsd:simpleType name="NumberOfTransactionsType">
1907         <xsd:restriction base="xsd:positiveInteger">
1908             <xsd:minInclusive value="1"/>
1909             <xsd:maxInclusive value="18446744073709551615"/>
1910         </xsd:restriction>
1911     </xsd:simpleType>
```


1912 There SHALL be only one <PermittedNumberOfTransactions> element within the <Permissions>
1913 element.

1914 The <PermittedNumberOfTransactions> element SHALL have one attribute named "any", that will have a
1915 "false" or "true" value, based on the following:

- 1916 • When the <PermittedNumberOfTransactions> element is null (i.e. it does not have any content in
1917 it), the value of the "any" attribute SHALL be set to "true" AND the XML Schema Instance (XSI) "nil"
1918 attribute SHALL be set to "true".
- 1919 • When the <PermittedNumberOfTransactions> element is not-null (i.e. it has a positive integer
1920 content in it), the value of the "any" attribute SHALL be set to "false" AND the XML Schema Instance
1921 (XSI) "nil" attribute SHALL NOT be present.

1922 A null <PermittedNumberOfTransactions> element specifies that applications are permitted use of the
1923 symmetric key for an unlimited number of encryption transactions, subject to complying with all other permission
1924 clauses in the <Permissions> element.

1925 The value of <PermittedNumberOfTransactions> element, of the XSD *positiveInteger* type, MAY range
1926 between 1 and 18446744073709551615.

1927 Some examples of the <PermittedNumberOfTransactions> element are shown below; other parts of their
1928 enclosing elements are not shown for brevity:

1929 **Example 1 – An example of a <PermittedNumberOfTransactions> element specifying that the**
1930 **symmetric key may be used only for a single encryption transaction by an SKCL:**

```
1931 <ekmi:PermittedNumberOfTransactions ekmi:any="false">  
1932 1  
1933 </ekmi:PermittedNumberOfTransactions>
```

1934 **Example 2 – An example of a <PermittedNumberOfTransactions> element specifying that the**
1935 **symmetric key may be used only for 100 transactions by an SKCL:**

```
1936 <ekmi:PermittedNumberOfTransactions ekmi:any="false">  
1937 100  
1938 </ekmi:PermittedNumberOfTransactions>
```

1939 **Example 3 – An example of a null <PermittedNumberOfTransactions> element specifying that the**
1940 **symmetric key may be used for an unlimited number of encryption transactions by an SKCL:**

```
1941 <ekmi:PermittedNumberOfTransactions ekmi:any="true" xsi:nil="true"/>
```

1942 2.20 Element <PermittedTimes> and <PermittedTime>

1943 The element <PermittedTimes>, of the type *PermittedTimesType* and its only child-element
1944 <PermittedTime>, which is an anonymous XSD *ComplexType*, are used to define sets of times during the day
1945 between which applications are permitted to use a symmetric key within a specific <Symkey> element.

1946 Schema Definition:

```
1947 <xsd:complexType name="PermittedTimesType">  
1948 <xsd:sequence>  
1949 <xsd:element name="PermittedTime" minOccurs="0" maxOccurs="unbounded">  
1950 <xsd:complexType>  
1951 <xsd:sequence>  
1952 <xsd:element name="StartTime">  
1953 <xsd:simpleType>  
1954 <xsd:restriction base="xsd:time">  
1955 <xsd:pattern value="\p{Nd}{2}:\p{Nd}{2}:\p{Nd}{2}"/>  
1956 </xsd:restriction>  
1957 </xsd:simpleType>
```

```

1958         </xsd:element>
1959         <xsd:element name="EndTime">
1960             <xsd:simpleType>
1961                 <xsd:restriction base="xsd:time">
1962                     <xsd:pattern value="\p{Nd}{2}:\p{Nd}{2}:\p{Nd}{2}"/>
1963                 </xsd:restriction>
1964             </xsd:simpleType>
1965         </xsd:element>
1966     </xsd:sequence>
1967 </xsd:complexType>
1968 </xsd:element>
1969 </xsd:sequence>
1970 <xsd:attribute ref="tns:any" use="required"/>
1971 </xsd:complexType>

```

1972 There SHALL be only one <PermittedTimes> element within the <Permissions> element. However, there
1973 MAY be an unbounded (unlimited) number of <PermittedTime> sub-elements within a <PermittedTimes>
1974 element.

1975 The <PermittedTimes> element SHALL have one attribute named "any", that will have a "false" or "true" value,
1976 based on the following:

- 1977 • When the <PermittedTimes> element is null (i.e. it does not have a single <PermittedTime> sub-
1978 element in it), the value of the "any" attribute SHALL be set to "true" AND the XML Schema Instance
1979 (XSI) "nil" attribute SHALL be set to "true".
- 1980 • When the <PermittedTimes> element is not-null (i.e. it has at least one <PermittedTime> sub-
1981 element in it), the value of the "any" attribute SHALL be set to "false" AND the XML Schema Instance
1982 (XSI) "nil" attribute SHALL NOT be present.

1983 A null <PermittedTimes> element specifies that applications are permitted use of the symmetric key at ANY
1984 time of the day or night, subject to complying with all other permission clauses in the <Permissions> element.

1985 The <PermittedTime> sub-element identifies an individual set of times between which application are
1986 permitted to use the symmetric key in question. The <PermittedTime> element consists of the following sub-
1987 elements:

- 1988 1. The <StartTime> element identifies the date from which applications may start using the symmetric
1989 key in question. It is an XSD *Time* type that MUST be specified in a specific pattern (see examples)
1990 where the first two digits specify the hour, the second two digits specify the minutes and the last two
1991 digits specify the seconds in a 24 hour format.

1992 There SHALL be only one <StartTime> element within a <PermittedTime> element.

1993 Conforming **SKCL** implementations SHALL NOT start using the symmetric before the onset of the
1994 <StartTime> on the client machine.

- 1997 2. The <EndTime> element identifies the time until which applications may use the symmetric key in
1998 question. It is an XSD *Time* type that MUST be specified in a specific pattern (see examples) where the
1999 first two digits specify the hour, the second two digits specify the minutes and the last two digits specify
2000 the seconds in a 24 hour format.

2001 There SHALL be only one <EndTime> element within a <PermittedTime> element.

2002 Conforming **SKCL** implementations SHALL NOT use the symmetric after the end of the <EndTime> on
2003 the client machine.

2004 Some examples of the <PermittedTimes> element are shown below; other parts of their enclosing elements
2005 are not shown for brevity:

2006 **Example 1 – An example of a <PermittedTimes> element with a single <PermittedTime> element. The**
2007 **<StartTime> specifies 9:00AM on the client machine while the <EndTime> specifies 5:00PM:**

```

2010     <ekmi:PermittedTimes ekmi:any="false">
2011         <ekmi:PermittedTime>
2012             <ekmi:StartTime>09:00:00</ekmi:StartTime>
2013             <ekmi:EndTime>17:00:00</ekmi:EndTime>
2014         </ekmi:PermittedTime>
2015     </ekmi:PermittedTimes>

```

2016 **Example 2 – An example of a <PermittedTimes> element with two <PermittedTime> elements. For the**
2017 **first <PermittedTime> element, the <StartTime> element specifies 6:00AM while the <EndTime>**
2018 **element specifies 12:00 Noon. For the second <PermittedTime> element, the <StartTime> element**
2019 **specifies 3:00 PM in the afternoon, while the <EndTime> element specifies 7:00PM in the evening. This**
2020 **policy might imply that a symmetric key with this <PermittedTimes> element cannot be used during a**
2021 **lunch break of 12:00 Noon to 3:00PM:**

```

2022     <ekmi:PermittedTimes ekmi:any="false">
2023         <ekmi:PermittedTime>
2024             <ekmi:StartTime>06:00:00</ekmi:StartTime>
2025             <ekmi:EndTime>12:00:00</ekmi:EndTime>
2026         </ekmi:PermittedTime>
2027         <ekmi:PermittedTime>
2028             <ekmi:StartTime>15:00:00</ekmi:StartTime>
2029             <ekmi:EndTime>19:00:00</ekmi:EndTime>
2030         </ekmi:PermittedTime>
2031     </ekmi:PermittedTimes>

```

2032 **Example 3 – An example of a null <PermittedTimes> element, specifying that the key may be used at**
2033 **any time:**

```

2034     <ekmi:PermittedTimes ekmi:any="true" xsi:nil="true"/>

```

2035 2.21 Element <PermittedUses> and <PermittedUse>

2036 The element <PermittedUses>, of the type *PermittedUsesType*, is used to define the specific ways in which
2037 applications are permitted to use a symmetric key within a specific <Symkey> element.

2038 Schema Definition:

```

2039     <xsd:complexType name="PermittedUsesType" mixed="true">
2040         <xsd:sequence>
2041             <xsd:element name="PermittedUse" minOccurs="0" maxOccurs="unbounded">
2042                 <xsd:simpleType>
2043                     <xsd:restriction base="xsd:string">
2044                         <xsd:maxLength value="256"/>
2045                         <xsd:whiteSpace value="preserve"/>
2046                     </xsd:restriction>
2047                 </xsd:simpleType>
2048             </xsd:element>
2049             <xsd:element name="Other" type="xsd:anyType" minOccurs="0"/>
2050         </xsd:sequence>
2051         <xsd:attribute ref="tns:any" use="required"/>
2052     </xsd:complexType>

```

2053 There SHALL be only one <PermittedUses> element within the <Permissions> element. However, there
2054 MAY be an unbounded (unlimited) number of <PermittedUse> sub-elements within the <PermittedUses>
2055 element.

2056 The <PermittedUses> element SHALL have one attribute named “any”, that will have a “false” or “true” value,
2057 based on the following:

- 2058 • When the <PermittedUses> element is null (i.e. it does not have a single <PermittedUse> sub-
2059 element in it), the value of the “any” attribute SHALL be set to “true” AND the XML Schema Instance
2060 (XSI) “nil” attribute SHALL be set to “true”.
- 2061 • When the <PermittedUses> element is not-null (i.e. it has at least one <PermittedUse> sub-
2062 element in it), the value of the “any” attribute SHALL be set to “false” AND the XML Schema Instance
2063 (XSI) “nil” attribute SHALL NOT be present.

2064 A null <PermittedUses> element specifies that applications are permitted use of the symmetric key for ANY
2065 purpose, subject to complying with all other permission clauses in the <Permissions> element.

2066 Examples of the <PermittedUses> element are shown below; other parts of their enclosing elements are not
2067 shown for brevity:

2068 **Example 1 – An example of a <PermittedUses> element specifying that the symmetric key may be used
2069 only by VPN applications for session encryption keys:**

```
2070 <ekmi:PermittedUses ekmi:any="false">
2071 <ekmi:PermittedUse>VPN</ekmi:PermittedUse>
2072 </ekmi:PermittedUses>
```

2073 **Example 2 – An example of a <PermittedUses> element specifying that the symmetric key may be used
2074 only by applications on laptops and Personal Digital Assistants (PDA):**

```
2075 <ekmi:PermittedUses ekmi:any="false">
2076 <ekmi:PermittedUse>Laptop</ekmi:PermittedUse>
2077 <ekmi:PermittedUse>PDA</ekmi:PermittedUse>
2078 </ekmi:PermittedUses>
```

2079 **Example 3 – An example of a null <PermittedUses> element specifying that the symmetric key may be
2080 used for any purpose:**

```
2081 <ekmi:PermittedUses ekmi:any="true" xsi:nil="true"/>
```

2082 2.22 Element <KeyCachePolicyRequest>

2083 The <KeyCachePolicyRequest> element is used to request a key-cache policy from the SKS server , so the
2084 client may know if and how to cache symmetric keys locally.

2085 While it is a top-level element within this specification, a <SymkeyRequest> element MUST be enclosed within a
2086 **SOAP Body** element of a **SOAP Envelope** to conform to the security requirements of this specification. The
2087 **SOAP Header** of the **SOAP Envelope** MUST enclose a **Security** element conforming to [WSS] with a
2088 **ValueType** attribute containing the value <http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-x509-token-profile-1.0#X509v3>. The **Security** element must conform to all other requirements of the specified security profile
2089 in [WSS] to form a well-formed, secure message.
2090

2091 Schema Definition:

```
2092 <xsd:element name="KeyCachePolicyRequest">
2093 <xsd:complexType>
2094 <xsd:annotation>
2095 <xsd:documentation>
2096 No elements/attributes are defined for KeyCachePolicyRequest.
2097 </xsd:documentation>
2098 </xsd:annotation>
2099 </xsd:complexType>
2100 </xsd:element>
```

2101 The <KeyCachePolicyRequest> has no child elements. The SOAP Header of the signed request provides the
2102 SKS server with all the information it needs to process the request: the identity of the requester, strong
2103 authentication and message integrity of the request.

2104 Some examples of the use of the <SymkeyRequest> element are as follows:

2105 **Example 1 – An example of a <KeyCachePolicyRequest>; the surrounding SOAP envelope is not**
2106 **displayed here for brevity:**

```
2107     <ekmi:KeyCachePolicyRequest  
2108         xmlns:ekmi="http://docs.oasis-open.org/ekmi/2008/01"/>
```

2109 **2.23 Element <KeyCachePolicyResponse>**

2110 The <KeyCachePolicyResponse> element is the response sent by an **SKS** Server to a client that requested a
2111 key-cache policy through a <KeyCachePolicyRequest>. The <KeyCachePolicyResponse> contains
2112 policy elements, which define rules that conforming implementations of the **SKCL** MUST adhere to when
2113 caching symmetric keys sent by the **SKS** Server.

2114 **Schema Definition:**

```
2115     <xsd:element name="KeyCachePolicyResponse">  
2116         <xsd:complexType>  
2117             <xsd:sequence>  
2118                 <xsd:element  
2119                     name="KeyCachePolicy"  
2120                     type="ekmi:KeyCachePolicyType"  
2121                     minOccurs="1" maxOccurs="unbounded"/>  
2122             </xsd:sequence>  
2123         </xsd:complexType>  
2124     </xsd:element>
```

2125 The <KeyCachePolicyResponse> element consists of a minimum of one, but an unbounded (unlimited)
2126 number of <KeyCachePolicy> children elements.

2127 **2.24 Element <KeyCachePolicy>**

2128 The <KeyCachePolicy> element contains policy elements, which define rules that conforming implementations
2129 of the **SKCL** MUST adhere to when caching symmetric keys sent by the **SKS** Server.

2130 **Schema Definition:**

```
2131     <xsd:element name="KeyCachePolicyResponse">  
2132         <xsd:complexType>  
2133             <xsd:sequence>  
2134                 <xsd:element  
2135                     name="KeyCachePolicy"  
2136                     type="ekmi:KeyCachePolicyType"  
2137                     minOccurs="1" maxOccurs="unbounded"/>  
2138             </xsd:sequence>  
2139         </xsd:complexType>  
2140     </xsd:element>  
  
2141     <xsd:complexType name="KeyCachePolicyType" mixed="true">  
2142         <xsd:sequence>  
2143             <xsd:element name="KeyCachePolicyID" type="tns:TwoPartIDType"/>  
2144             <xsd:element name="PolicyName">  
2145                 <xsd:simpleType>  
2146                     <xsd:restriction base="xsd:string">  
2147                         <xsd:maxLength value="255"/>  
2148                         <xsd:whiteSpace value="preserve"/>  
2149                     </xsd:restriction>  
2150                 </xsd:simpleType>  
2151             </xsd:sequence>
```

```

2152     <xsd:element name="Description" nillable="true">
2153         <xsd:simpleType>
2154             <xsd:restriction base="xsd:string">
2155                 <xsd:maxLength value="2048"/>
2156                 <xsd:whiteSpace value="preserve"/>
2157             </xsd:restriction>
2158         </xsd:simpleType>
2159     </xsd:element>
2160     <xsd:element name="KeyClass" type="tns:KeyClassType"/>
2161     <xsd:element name="StartDate" type="xsd:dateTime"/>
2162     <xsd:element name="EndDate" type="xsd:dateTime" nillable="true"/>
2163     <xsd:element name="PolicyCheckInterval">
2164         <xsd:simpleType>
2165             <xsd:restriction base="xsd:nonNegativeInteger">
2166                 <xsd:minInclusive value="0"/>
2167                 <xsd:maxInclusive value="2592000"/>
2168             </xsd:restriction>
2169         </xsd:simpleType>
2170     </xsd:element>
2171     <xsd:element name="Status" type="tns:StatusType"/>
2172     <xsd:element
2173         name="NewKeysCacheDetail"
2174         type="tns:KeyCacheDetailType"
2175         minOccurs="0"/>
2176     <xsd:element
2177         name="UsedKeysCacheDetail"
2178         type="tns:KeyCacheDetailType"
2179         minOccurs="0"/>
2180     </xsd:sequence>
2181 </xsd:complexType>

```

2182 The <KeyCachePolicy> element is of the **KeyCachePolicyType** and consists of the following child elements:

2183 1. <KeyCachePolicyID> [Required]

2184 The <KeyCachePolicyID> element, of type **TwoPartIDType**, identifies the unique policy object within
2185 the **SKMS**. There SHALL be only one <KeyCachePolicyID> element within a <KeyCachePolicy>
2186 element.
2187

2188 The **TwoPartIDType** is specified in Section 2.8.
2189

2190 2. <PolicyName> [Required]

2191 The <PolicyName> element, of type XSD **String**, with a maximum length of 255 characters, identifies
2192 a unique name given to this <KeyCachePolicy>. There SHALL be only one <PolicyName> element
2193 within a <KeyCachePolicy> element.
2194

2195 3. <Description> [Required]

2196 The <Description> element, of type XSD **String**, with a maximum length of 2048 characters,
2197 provides a human-readable description of this policy. There SHALL be only one <Description>
2198 element within a <KeyCachePolicy> element.
2199

2200 The <Description> MAY be an empty element, but MUST exist within the <KeyCachePolicy>
2201 element.
2202

2203 4. <KeyClass> [Required]

2204 This element of type **KeyClassType** identifies the key-class of the symmetric key to which this policy
2205 applies.
2206

- 2207 5. <StartDate> [Required]
2208
2209 The <StartDate> element , of type XSD *dateTime*, specifies the date and time at which this policy
2210 becomes effective. There SHALL be only one <StartDate> element within a <KeyCachePolicy>
2211 element.
- 2212 6. <EndDate> [Required]
2213
2214 The <EndDate> element , of type XSD *dateTime*, specifies the date and time at which this policy
2215 expires. There SHALL be only one <EndDate> element within a <KeyCachePolicy> element.
2216
2217 The <EndDate> MAY be an empty element, but MUST exist within the <KeyCachePolicy> element.
- 2218 7. <PolicyCheckInterval> [Required]
2219
2220 The <PolicyCheckInterval> element , of type XSD *nonNegativeInteger*, specifies the frequency at
2221 which the client SHALL check the SKS server for updates to this policy. This frequency is specified in
2222 seconds and SHALL NOT exceed 2592000 seconds (30 calendar days). There SHALL be only one
2223 <PolicyCheckInterval> element within a <KeyCachePolicy> element.
- 2224 8. <Status> [Required]
2225
2226 The <Status> element, of type *StatusType*, identifies the current status of this policy within the SKMS.
2227 There SHALL be only one <Status> element within a <KeyCachePolicy> element.
2228
2229 The *StatusType* is specified in Section 2.11.
- 2230 9. <NewKeysCacheDetail> [Required]
2231
2232 The <NewKeysCacheDetail> element, of type *KeyCacheDetailType*, defines how many new (as yet
2233 unused for any encryption transaction) symmetric keys a client may cache, and for how long. It is the
2234 responsibility of the conforming SKCL implementation to enforce these rules.
2235
2236 The absence of the <NewKeysCacheDetail> element implies that new symmetric keys SHALL NEVER
2237 be cached on the client. New keys may be cached only when this element exists, and SHALL conform
2238 to the rules specified in this element.
2239
2240 When it exists, there SHALL be only one <NewKeysCacheDetail> element in a <KeyCachePolicy>
2241 element.
2242
2243 The *KeyCacheDetailType* is specified in Section 2.22.
- 2244 10. <UsedKeysCacheDetail> [Required]
2245
2246 The <UsedKeysCacheDetail> element, of type *KeyCacheDetailType*, defines how many used
2247 symmetric keys a client may cache, and for how long. It is the responsibility of the conforming SKCL
2248 implementation to enforce these rules.
2249
2250 The absence of the <UsedKeysCacheDetail> element implies that used symmetric keys SHALL
2251 NEVER be cached on the client. Used keys may be cached only when this element exists, and SHALL
2252 conform to the rules specified in this element.
2253
2254 When it exists, there SHALL be only one <UsedKeysCacheDetail> element in a
2255 <KeyCachePolicy> element.
2256
2257 The *KeyCacheDetailType* is specified in Section 2.22.
- 2258 Some examples of the <KeyUsePolicy> element are as follows.

2259 **Example 1 – A <KeyCachePolicy> that is valid between January 01, 2008 and December 31, 2008. It**
2260 **requires the client to check for policy updates every day and allows 3 new and 3 used keys to be cached**
2261 **for up to 90 days:**

```
2262     <ekmi:KeyCachePolicy>
2263         <ekmi:KeyCachePolicyID>10514-17</ekmi:KeyCachePolicyID>
2264         <ekmi:PolicyName>
2265             Corporate Laptop Symmetric Key Caching Policy
2266         </ekmi:PolicyName>
2267         <ekmi:Description>
2268             This policy defines how company-issued laptops will manage
2269             symmetric keys used for file/disk encryption in each laptop's
2270             local cache. This policy must be used by all laptops that
2271             use the company EKMI.
2272         </ekmi:Description>
2273         <ekmi:StartDate>2008-01-01T00:00:01.0</ekmi:StartDate>
2274         <ekmi:EndDate>2008-12-31T24:00:00.0</ekmi:EndDate>
2275         <ekmi:PolicyCheckInterval>86400</ekmi:PolicyCheckInterval>
2276         <ekmi:Status>Active</ekmi:Status>
2277         <ekmi:NewKeysCacheDetail>
2278             <ekmi:MaximumKeys>3</ekmi:MaximumKeys>
2279             <ekmi:MaximumDuration>7776000</ekmi:MaximumDuration>
2280         </ekmi:NewKeysCacheDetail>
2281         <ekmi:UsedKeysCacheDetail>
2282             <ekmi:MaximumKeys>3</ekmi:MaximumKeys>
2283             <ekmi:MaximumDuration>7776000</ekmi:MaximumDuration>
2284         </ekmi:UsedKeysCacheDetail>
2285     </ekmi:KeyCachePolicy>
```

2286 **Example 2 – A <KeyCachePolicy> that is effective starting January 01, 2008 and never expires. It does**
2287 **NOT permit any caching of symmetric keys through the absence of the detail elements on caching:**

```
2288     <ekmi:KeyCachePolicy>
2289         <ekmi:KeyCachePolicyID>10514-1</ekmi:KeyCachePolicyID>
2290         <ekmi:PolicyName>
2291             No Caching Policy
2292         </ekmi:PolicyName>
2293         <ekmi:Description>
2294             This policy is for high-risk, always-connected machines on the
2295             network, which will never cache symmetric keys locally. This
2296             policy never expires (but checks monthly for any updates).
2297         </ekmi:Description>
2298         <ekmi:StartDate>2008-01-01T00:00:01.0</ekmi:StartDate>
2299         <ekmi:EndDate>1969-01-01T00:00:00.0</ekmi:EndDate>
2300         <ekmi:PolicyCheckInterval>2592000</ekmi:PolicyCheckInterval>
2301         <ekmi:Status>Active</ekmi:Status>
2302     </ekmi:KeyCachePolicy>
```

2303 **2.25 Type *KeyCacheDetailType***

2304 The ***KeyCacheDetailType*** type allows SKS servers to specify precisely how many symmetric keys MAY be
2305 cached on the client machine, and for how long.

2306 **Schema Definition:**

```
2307     <xsd:complexType name="KeyCacheDetailType">
2308         <xsd:sequence>
2309             <xsd:element name="MaximumKeys" minOccurs="1">
2310                 <xsd:simpleType>
2311                     <xsd:restriction base="xsd:integer">
```



```

2312         <xsd:minInclusive value="0"/>
2313         <xsd:maxInclusive value="18446744073709551615"/>
2314     </xsd:restriction>
2315 </xsd:simpleType>
2316 </xsd:element>
2317 <xsd:element name="MaximumDuration" minOccurs="1">
2318     <xsd:simpleType>
2319         <xsd:restriction base="xsd:integer">
2320             <xsd:minInclusive value="0"/>
2321             <xsd:maxInclusive value="18446744073709551615"/>
2322         </xsd:restriction>
2323     </xsd:simpleType>
2324 </xsd:element>
2325 </xsd:sequence>
2326 </xsd:complexType>

```

2327 The **KeyCacheDetailType** consists of the following child elements:

2328 1. <MaximumKeys> [Required]

2329

2330 The <MaximumKeys> element, of type XSD **Integer**, specifies the maximum number of symmetric keys
2331 that MAY be cached on a client machine. It SHALL be a positive number between the values 0 and
2332 18446744073709551615. There SHALL be only one <MaximumKeys> element within an element that
2333 uses the **KeyCacheDetailType**.

2334 2. <MaximumDuration> [Required]

2335

2336 The <MaximumDuration> element, of type XSD **Integer**, specifies the maximum number of seconds
2337 that symmetric keys MAY be cached on a client machine. It SHALL be a positive number between the
2338 values 0 and 18446744073709551615. There SHALL be only one <MaximumDuration> element
2339 within an element that uses the **KeyCacheDetailType**.

2340 Examples of the **KeyCacheDetailType** when used in the <KeyCachePolicy> element are as follows.

2341 **Example 1 – A <KeyCachePolicy> that is valid between January 01, 2008 and December 31, 2008. It**
2342 **requires the client to check for policy updates every day and allows 3 new and 3 used keys to be cached**
2343 **for up to 90 days:**

```

2344 <ekmi:KeyCachePolicy>
2345     <ekmi:KeyCachePolicyID>10514-17</ekmi:KeyCachePolicyID>
2346     <ekmi:PolicyName>
2347         Corporate Laptop Symmetric Key Caching Policy
2348     </ekmi:PolicyName>
2349     <ekmi:Description>
2350         This policy defines how company-issued laptops will manage
2351         symmetric keys used for file/disk encryption in their local
2352         cache. This policy must be used by all laptops that use
2353         the company EKMI.
2354     </ekmi:Description>
2355     <ekmi:StartDate>2008-01-01T00:00:01.0</ekmi:StartDate>
2356     <ekmi:EndDate>2008-12-31T24:00:00.0</ekmi:EndDate>
2357     <ekmi:PolicyCheckInterval>86400</ekmi:PolicyCheckInterval>
2358     <ekmi:Status>Active</ekmi:Status>
2359     <ekmi:NewKeysCacheDetail>
2360         <ekmi:MaximumKeys>3</ekmi:MaximumKeys>
2361         <ekmi:MaximumDuration>7776000</ekmi:MaximumDuration>
2362     </ekmi:NewKeysCacheDetail>
2363     <ekmi:UsedKeysCacheDetail>
2364         <ekmi:MaximumKeys>3</ekmi:MaximumKeys>
2365         <ekmi:MaximumDuration>7776000</ekmi:MaximumDuration>

```

2366 </ekmi:UsedKeysCacheDetail>
2367 </ekmi:KeyCachePolicy>

2368 **Example 1 – A <KeyCachePolicy> that is valid between January 01, 2008 and December 31, 2008. It**
2369 **requires the client to check for policy updates every day and allows 1 new and 0 used keys to be cached**
2370 **for upto 15 days:**

```
2371       <ekmi:KeyCachePolicy>
2372        <ekmi:KeyCachePolicyID>10514-17</ekmi:KeyCachePolicyID>
2373        <ekmi:PolicyName>
2374          Corporate Laptop Symmetric Key Caching Policy
2375        </ekmi:PolicyName>
2376        <ekmi:Description>
2377          This policy defines how company-issued laptops will manage
2378          symmetric keys used for file/disk encryption in each laptop's
2379          local cache. This policy must be used by all laptops that
2380          use the company EKMI.
2381        </ekmi:Description>
2382        <ekmi:StartDate>2008-01-01T00:00:01.0</ekmi:StartDate>
2383        <ekmi:EndDate>2008-12-31T24:00:00.0</ekmi:EndDate>
2384        <ekmi:PolicyCheckInterval>86400</ekmi:PolicyCheckInterval>
2385        <ekmi>Status>Active</ekmi>Status>
2386        <ekmi>NewKeysCacheDetail>
2387          <ekmi:MaximumKeys>1</ekmi:MaximumKeys>
2388          <ekmi:MaximumDuration>1296000</ekmi:MaximumDuration>
2389        </ekmi>NewKeysCacheDetail>
2390        <ekmi:UsedKeysCacheDetail>
2391          <ekmi:MaximumKeys>0</ekmi:MaximumKeys>
2392          <ekmi:MaximumDuration>1296000</ekmi:MaximumDuration>
2393        </ekmi:UsedKeysCacheDetail>
2394        </ekmi:KeyCachePolicy>
```

2395 **Appendix A. Acknowledgments**

2396 The following individuals have participated in the creation of this specification and are gratefully
2397 acknowledged

2398 **Participants:**

2399 •

2400

2401

Appendix B. Revision History

Version	Date	Author	Notes
DRAFT 4	June 08, 2008	Arshad Noor	Initial version
DRAFT 5	June 17, 2008	Arshad Noor	Moved non-normative sections to their own document. KeyClass element was added to KeyCachePolicy. KeyCachePolicy is now embedded inside a KeyCachePolicyResponse.

2402

2403

2404

2405

Appendix C. Non-Normative Text

2406

Appendix D. SKSML Error Codes and Error Messages