



[Symmetric Key Services Markup Language (SKSML) Version 1.0 Normative DRAFT 5.1]

OASIS Technical Committee DRAFT

19 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>.....	14
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>.....	28
116	2.14 Element <PermittedDates> and <PermittedDate>.....	30
117	2.15 Element <PermittedDays> and <PermittedDay>.....	32
118	2.16 Element <PermittedDuration>.....	33
119	2.17 Element <PermittedLevels> and <PermittedLevel>.....	34
120	2.18 Element <PermittedLocations> and <PermittedLocation>.....	35
121	2.19 Element <PermittedNumberOfTransactions>.....	37
122	2.20 Element <PermittedTimes> and <PermittedTime>.....	38
123	2.21 Element <PermittedUses> and <PermittedUse>.....	39
124	2.22 Element <KeyCachePolicyRequest>.....	40
125	2.23 Element <KeyCachePolicyResponse>.....	41
126	2.24 Element <KeyCachePolicy>.....	41
127	2.25 Type KeyCacheDetailType.....	44
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/xmldsig-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 (SMIPv2)*.
204 IETF RFC 2578, April 1999. <http://www.ietf.org/html/rfc2578>

205

206

207

2 Specification

208

2.1 Element <SymkeyRequest>

209
210
211

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.

212
213
214
215
216
217

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.

218

Schema Definition:

219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235

```
<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>
```

236

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

237
238

1. <GlobalKeyID> [Required]

239
240
241
242
243
244

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.

245
246

2. <KeyClasses> [Optional]

247
248
249
250
251
252

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.

253
254
255

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.

256

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> elements
258 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 consisting
319 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-zero.
338 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-zero.
349 It will typically contain the unique key identifier of the symmetric key within the SKS server where the key was
350 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 an
357 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 <KeyClasses>
401 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. (Examples
407 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
477 This element of type **SymkeyType**, is returned by the **SKS** server in response to a successful
478 processing of a <SymkeyRequest>. There **MAY** be more than one <Symkey> element in the
479 <SymkeyResponse> if the client application made a request for multiple symmetric keys.
480

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

482 2. <SymkeyError> [Optional]

483
484 This element of type **SymkeyErrorType**, contains a response to a failed attempt in processing a
485 request for one or more symmetric keys. There **MAY** be more than one <SymkeyError> element in the
486 <SymkeyResponse> if the client application made a request for multiple symmetric keys and the request
487 resulted in multiple errors.
488

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

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 request for nine symmetric keys, each of a different key class:**

```
515 <ekmi:SymkeyRequest
516   xmlns:ekmi="http://docs.oasis-open.org/ekmi/2008/01">
517   <ekmi:GlobalKeyID>10514-0-0</ekmi:GlobalKeyID>
518   <ekmi:KeyClasses>
519     <ekmi:KeyClass>EHR-CDC</ekmi:KeyClass>
520     <ekmi:KeyClass>EHR-CRO</ekmi:KeyClass>
521     <ekmi:KeyClass>EHR-DEF</ekmi:KeyClass>
522     <ekmi:KeyClass>EHR-EMT</ekmi:KeyClass>
523     <ekmi:KeyClass>EHR-HOS</ekmi:KeyClass>
524     <ekmi:KeyClass>EHR-INS</ekmi:KeyClass>
525     <ekmi:KeyClass>EHR-NUR</ekmi:KeyClass>
526     <ekmi:KeyClass>EHR-PAT</ekmi:KeyClass>
527     <ekmi:KeyClass>EHR-PHY</ekmi:KeyClass>
528   </ekmi:KeyClasses>
529 </ekmi:SymkeyRequest>
```

530 **Example 6 – A response with one symmetric key and one error:**

```
531 <ekmi:SymkeyResponse
532   xmlns:ekmi="http://docs.oasis-open.org/ekmi/2008/01">
533   <ekmi:Symkey>.....</ekmi:Symkey>
534   <ekmi:SymkeyError>.....</ekmi:SymkeyError>
535 </ekmi:SymkeyResponse>
```

536 **Example 7 – A response with multiple symmetric keys and multiple error:**

```
537 <ekmi:SymkeyResponse
538   xmlns:ekmi="http://docs.oasis-open.org/ekmi/2008/01">
539   <ekmi:Symkey>.....</ekmi:Symkey>
540   <ekmi:Symkey>.....</ekmi:Symkey>
541   <ekmi:Symkey>.....</ekmi:Symkey>
542   <ekmi:SymkeyError>.....</ekmi:SymkeyError>
543   <ekmi:SymkeyError>.....</ekmi:SymkeyError>
544 </ekmi:SymkeyResponse>
```

545 2.5 Element <Symkey>

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

549 Schema Definition:

```
550 <xsd:complexType name="SymkeyType">  
551 <xsd:sequence>  
552 <xsd:element name="GlobalKeyID" type="ekmi:GlobalKeyIDType"/>  
553 <xsd:element name="KeyUsePolicy" type="ekmi:KeyUsePolicyType"/>  
554 <xsd:element name="EncryptionMethod" type="xenc:EncryptionMethodType"/>  
555 <xsd:element ref="xenc:CipherData"/>  
556 </xsd:sequence>  
557 </xsd:complexType>
```

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

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

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

564 1. <GlobalKeyID> [Required]

565
566 This element of type *GlobalKeyIDType* identifies the unique identifier of the symmetric key within an
567 SKMS. There SHALL be only one <GlobalKeyID> within a <Symkey> element.

568 The *GlobalKeyIDType* is specified in Section 2.2.

570 2. <KeyUsePolicy> [Required]

571
572 This element of type *KeyUsePolicyType*, defines how the symmetric key in this <Symkey> element may
573 be used by applications. There SHALL be only one <KeyUsePolicy> element within a <Symkey>
574 element.

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

577 3. <EncryptionMethod> [Required]

578
579 This element of type *EncryptionMethodType* from [XMLEncryption] describes how the symmetric key
580 in this <Symkey> element is encrypted for transport between the SKS Server and the client application.

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

- 583
584 - http://www.w3.org/2001/04/xmlenc#rsa-1_5
585 - <http://www.w3.org/2001/04/xmlenc#rsa-oaep-mgf1p>
586

587 4. <CipherData> [Required]

588
589 This element of *CipherDataType* from [XMLEncryption] contains the encrypted symmetric-key. As
590 specified in [XMLEncryption], the content of this element is Base-64 encoded and is of the XML Schema
591 *base64Binary* type.

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

594 Example 1 – A response with a symmetric key:

```

595 <ekmi:SymkeyResponse
596     xmlns:ekmi="http://docs.oasis-open.org/ekmi/2008/01">
597     <ekmi:Symkey>
598         <ekmi:GlobalKeyID>10514-1-235</ekmi:GlobalKeyID>
599         <ekmi:KeyUsePolicy>.....</ekmi:KeyUsePolicy>
600         <ekmi:EncryptionMethod
601             Algorithm="http://www.w3.org/2001/04/xmlenc#rsa-1_5"/>
602         <xenc:CipherData>
603             <xenc:CipherValue>
604                 E9zWB/y93hVSzeTLiDcQoDxmlNxTux+SffMNwCJmt1dIqzQHBnpdQ8
605                 1g6DKdkCFjJMhQhywCx9sfYjv9h5FDqUiQXG0ca8EU871zBoXBjDxj
606                 fg1pU8tGFbpWZcd/ATpJD/UJow/qimxi8+huUYJMtaGH=
607             </xenc:CipherValue>
608         </xenc:CipherData>
609     </ekmi:Symkey>
610 </ekmi:SymkeyResponse>

```

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

```

612 <ekmi:SymkeyResponse
613     xmlns:ekmi="http://docs.oasis-open.org/ekmi/2008/01">
614     <ekmi:Symkey>
615         <ekmi:GlobalKeyID>10514-1-235</ekmi:GlobalKeyID>
616         <ekmi:KeyUsePolicy>.....</ekmi:KeyUsePolicy>
617         <ekmi:EncryptionMethod
618             Algorithm="http://www.w3.org/2001/04/xmlenc#rsa-oaep-mgf1p"/>
619         <xenc:CipherData>
620             <xenc:CipherValue>
621                 E9zWB/y93hVSzeTLiDcQoDxmlNxTux+SffMNwCJmt1dIqzQHBnpdQ8
622                 1g6DKdkCFjJMhQhywCx9sfYjv9h5FDqUiQXG0ca8EU871zBoXBjDxj
623                 fg1pU8tGFbpWZcd/ATpJD/UJow/qimxi8+huUYJMtaGH=
624             </xenc:CipherValue>
625         </xenc:CipherData>
626     </ekmi:Symkey>
627     <ekmi:Symkey>
628         <ekmi:GlobalKeyID>10514-1-236</ekmi:GlobalKeyID>
629         <ekmi:KeyUsePolicy>.....</ekmi:KeyUsePolicy>
630         <ekmi:EncryptionMethod
631             Algorithm="http://www.w3.org/2001/04/xmlenc#rsa-oaep-mgf1p"/>
632         <xenc:CipherData>
633             <xenc:CipherValue>
634                 Qbg65cy93hVSzeTLiDcQoDxmlNxTux+SffMNwCJmt1dIqzQHBnpdQ8
635                 7k6DKdkCFjJMhQhywCx9sfYjv9h5FDqUiQXG0ca8EU871zBoXBjDxj
636                 uyecU8tGFbpWZcd/ATpJD/UJow/qimxi8+huUYJMtaGH=
637             </xenc:CipherValue>
638         </xenc:CipherData>
639     </ekmi:Symkey>
640 </ekmi:SymkeyResponse>

```

641 **2.6 Element <SymkeyError>**

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

644 **Schema Definition:**

```

645 <xsd:complexType name="SymkeyErrorType">
646     <xsd:sequence>
647         <xsd:element name="RequestedGlobalKeyID" type="ekmi:GlobalKeyIDType"/>
648     </xsd:sequence>

```

```

649         name="RequestedKeyClass"
650         type="ekmi:KeyClassType"
651         minOccurs="0"/>
652     <xsd:element name="ErrorCode">
653         <xsd:simpleType>
654             <xsd:restriction base="xsd:string">
655                 <xsd:maxLength value="255"/>
656             </xsd:restriction>
657         </xsd:simpleType>
658     </xsd:element>
659     <xsd:element name="ErrorMessage">
660         <xsd:simpleType>
661             <xsd:restriction base="xsd:string">
662                 <xsd:maxLength value="1024"/>
663             </xsd:restriction>
664         </xsd:simpleType>
665     </xsd:element>
666 </xsd:sequence>
667 </xsd:complexType>

```

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

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

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

675 1. <RequestedGlobalKeyID> [Required]

676
677 This element of type **GlobalKeyIDType** identifies the unique identifier of the symmetric key requested by
678 the client application. There **SHALL** be only one <RequestedGlobalKeyID> within a <SymkeyError>
679 element.

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

682 2. <RequestedKeyClass> [Optional]

683
684 This element of type **KeyClassType** identifies the key-class of the symmetric key requested by the
685 client application. If the <RequestedKeyClass> element is not embedded in the <SymkeyError>
686 element, this implies that the requested symmetric key was for the default key-class of the SKMS.

687 The **KeyClassType** is specified in Section 2.3.

689 3. <ErrorCode> [Required]

690
691 This element of type **String** identifies a mnemonic code identifying the error the **SKS** Server
692 experienced in processing the client's symmetric key request.

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

695 4. <ErrorMessage> [Required]

696
697 This element of type **String** identifies a localized message describing the error the **SKS** Server
698 experienced in processing the client's symmetric key request.

699 The <ErrorMessage> element **SHALL** return the appropriate localized version of the message
700 corresponding to the <ErrorCode> element from Appendix D of this specification.

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

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

```
704 <ekmi:SymkeyResponse
705     xmlns:ekmi="http://docs.oasis-open.org/ekmi/2008/01">
706     <ekmi:SymkeyError>
707         <ekmi:RequestedGlobalKeyID>10514-2-22</ekmi:RequestedGlobalKeyID>
708         <ekmi:ErrorCode>SKS-100004</ekmi:ErrorCode>
709         <ekmi:ErrorMessage>Unauthorized request for key</ekmi:ErrorMessage>
710     </ekmi:SymkeyError>
711 </ekmi:SymkeyResponse>
```

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

```
713 <ekmi:SymkeyResponse
714     xmlns:ekmi="http://docs.oasis-open.org/ekmi/2008/01">
715     <ekmi:SymkeyError>
716         <ekmi:RequestedGlobalKeyID>10514-1-0</ekmi:RequestedGlobalKeyID>
717         <ekmi:ErrorCode>SKS-100001</ekmi:ErrorCode>
718         <ekmi:ErrorMessage>Invalid GlobalKeyID</ekmi:ErrorMessage>
719     </ekmi:SymkeyError>
720     <ekmi:SymkeyError>
721         <ekmi:RequestedGlobalKeyID>10514-2-22</ekmi:RequestedGlobalKeyID>
722         <ekmi:ErrorCode>SKS-100004</ekmi:ErrorCode>
723         <ekmi:ErrorMessage>Unauthorized request for key</ekmi:ErrorMessage>
724     </ekmi:SymkeyError>
725 </ekmi:SymkeyResponse>
```

726 **2.7 Element <KeyUsePolicy>**

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

729 **Schema Definition:**

```
730 <xsd:complexType name="KeyUsePolicyType" mixed="true">
731     <xsd:sequence>
732         <xsd:element name="KeyUsePolicyID" type="tns:TwoPartIDType"/>
733         <xsd:element name="PolicyName">
734             <xsd:simpleType>
735                 <xsd:restriction base="xsd:string">
736                     <xsd:maxLength value="255"/>
737                 </xsd:restriction>
738             </xsd:simpleType>
739         </xsd:element>
740         <xsd:element name="KeyClass" type="tns:KeyClassType"/>
741         <xsd:element name="KeyAlgorithm" type="tns:EncryptionAlgorithmType"/>
742         <xsd:element name="KeySize" type="tns:KeySizeType"/>
743         <xsd:element name="Status" type="tns:StatusType"/>
744         <xsd:element name="Permissions" type="tns:PermissionsType"/>
745     </xsd:sequence>
746 </xsd:complexType>
```

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

748 1. <KeyUsePolicyID> [Required]

749
750 The <KeyUsePolicyID> element, of type **TwoPartIDType** , identifies the unique policy object within
751 the SKMS. There SHALL be only one <KeyUsePolicyID> element within a <KeyUsePolicy> element.

752
753 The **TwoPartIDType** is specified in Section 2.8.

- 754 2. <PolicyName> [Required]
755
756 The <PolicyName> element, of type XSD *String*, with a maximum length of 255 characters,
757 identifies a unique name given to this <KeyUsePolicy>. There SHALL be only one <PolicyName>
758 element within a <KeyUsePolicy> element.
- 759 3. <KeyClass> [Required]
760
761 The <KeyClass> element, of type *KeyClassType*, identifies a key-class assigned to this
762 <KeyUsePolicy>. There SHALL be only one <KeyUsePolicyID> element within a <KeyUsePolicy>
763 element.
764
765 The *KeyClassType* is specified in Section 2.3.
- 766 4. <KeyAlgorithm> [Required]
767
768 The <KeyAlgorithm> element, of type *EncryptionAlgorithmType*, identifies encryption algorithm to
769 be used by applications when using this symmetric key. There SHALL be only one <KeyAlgorithm>
770 element within a <KeyUsePolicy> element.
771
772 The <KeyAlgorithm> element is specified in Section 2.9.
- 773 5. <KeySize> [Required]
774
775 The <KeySize> element, of type *KeySizeType*, defines the size of the symmetric key, in bits (binary
776 digits). There SHALL be only one <KeySize> element within a <KeyUsePolicy> element.
777
778 Note: It is possible to determine the size of a symmetric key in an **SKCL** implementation without having
779 to send the size in the response. So, why include it? It is our belief that while network bandwidth and
780 compute performance of devices are increasing steadily, encryption is desired in many small and
781 portable devices. Consequently, it will speed up applications in cryptographic processing if they do not
782 have to determine the size of each key they use. While “protocol purity” demands that implementation
783 issues do not show up in protocol design, we believe it is justified in this case.
784
785 The *KeySizeType* is specified in Section 2.10.
- 786 6. <Status> [Required]
787
788 The <Status> element, of type *StatusType*, identifies the current status of the symmetric key. There
789 SHALL be only one <Status> element within a <KeyUsePolicy> element.
790
791 The *StatusType* is specified in Section 2.11.
- 792 7. <Permissions> [Required]
793
794 The <Permissions> element, of type *PermissionsType*, defines what is permissible to client
795 applications with the symmetric key this element is associated with. It is the responsibility of the
796 conforming **SKCL** implementation to enforce these rules.
797
798 An important distinction of this element – unlike most access control rules – is that the absence of sub-
799 elements in the <Permissions> element implies that all permissions are allowed. The presence of
800 sub-elements in this element provide rules to the **SKCL** about what actions are permitted.
801
802 There SHALL be only one <Permissions> element in a <KeyUsePolicy> element.
803
804 The *PermissionsType* is specified in Section 2.12.
- 805 Some examples of the <KeyUsePolicy> element are as follows.
- 806 **Example 1 – A <KeyUsePolicy> with some permissions:**

```

807 <ekmi:KeyUsePolicy>
808   <ekmi:KeyUsePolicyID>10514-4</ekmi:KeyUsePolicyID>
809   <ekmi:PolicyName>DES-EDE KeyUsePolicy</ekmi:PolicyName>
810   <ekmi:KeyClass>HR-Class</ekmi:KeyClass>
811   <ekmi:KeyAlgorithm>
812     http://www.w3.org/2001/04/xmlenc#tripleDES-cbc
813   </ekmi:KeyAlgorithm>
814   <ekmi:KeySize>192</ekmi:KeySize>
815   <ekmi:Status>Active</ekmi:Status>
816   <ekmi:Permissions>
817     <ekmi:PermittedApplications>
818       <ekmi:PermittedApplication>
819         <ekmi:ApplicationID>10514-23</ekmi:ApplicationID>
820         <ekmi:ApplicationName>Payroll Application</ekmi:ApplicationName>
821         <ekmi:Version>1.0</ekmi:Version>
822         <ekmi:DigestAlgorithm>
823           http://www.w3.org/2000/09/xmldsig#sha1
824         </ekmi:DigestAlgorithm>
825         <ekmi:DigestValue>NIG4bKkt4cziEqFFu0oBTM81efU=</ekmi:DigestValue>
826       </ekmi:PermittedApplication>
827     </ekmi:PermittedApplications>
828     <ekmi:PermittedTimes>
829       <ekmi:PermittedTime>
830         <ekmi:StartTime>07:00:00</ekmi:StartTime>
831         <ekmi:EndTime>19:00:00</ekmi:EndTime>
832       </ekmi:PermittedTime>
833     </ekmi:PermittedTimes>
834   </ekmi:Permissions>
835 </ekmi:KeyUsePolicy>

```

836 **Example 2 – A <KeyUsePolicy> with some permissions:**

```

837 <ekmi:KeyUsePolicy>
838   <ekmi:KeyUsePolicyID>10514-2</ekmi:KeyUsePolicyID>
839   <ekmi:PolicyName>Laptop KeyUsePolicy</ekmi:PolicyName>
840   <ekmi:KeyClass>HR-Class</ekmi:KeyClass>
841   <ekmi:KeyAlgorithm>
842     http://www.w3.org/2001/04/xmlenc#aes256-cbc
843   </ekmi:KeyAlgorithm>
844   <ekmi:KeySize>256</ekmi:KeySize>
845   <ekmi:Status>Active</ekmi:Status>
846   <ekmi:Permissions/>
847 </ekmi:KeyUsePolicy>

```

848 2.8 Type *TwoPartIDType*

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

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

852 Schema Definition:

```

853 <xsd:simpleType name="TwoPartIDType">
854   <xsd:restriction base="xsd:string">
855     <xsd:minLength value="3"/>
856     <xsd:maxLength value="41"/>
857     <xsd:pattern value="[1-9][0-9]{0,19}-[1-9][0-9]{0,19}"/>
858     <xsd:whiteSpace value="collapse"/>

```

```
859     </xsd:restriction>
860 </xsd:simpleType>
```

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

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

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

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

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

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

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

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

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

```
871     <ekmi:ApplicationID>
872         18446744073709551615-18446744073709551615
873 </ekmi:ApplicationID>
```

874 2.9 Element <KeyAlgorithm>

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

877 **Schema Definition:**

```
878     <xsd:simpleType name="EncryptionAlgorithmType">
879         <xsd:restriction base="xsd:anyURI">
880             <xsd:enumeration value="http://www.w3.org/2001/04/xmlenc#tripledes-cbc"/>
881             <xsd:enumeration value="http://www.w3.org/2001/04/xmlenc#aes128-cbc"/>
882             <xsd:enumeration value="http://www.w3.org/2001/04/xmlenc#aes192-cbc"/>
883             <xsd:enumeration value="http://www.w3.org/2001/04/xmlenc#aes256-cbc"/>
884         </xsd:restriction>
885     </xsd:simpleType>
```

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

889 1. Triple Data Encryption Standard (3DES)

890

891 Within the context of this specification, and as specified in [XMLEncryption], the form of 3DES
892 supported within SKSML is a 192-bit key with a 64-bit Initialization Vector. Of the key bits, the first 64
893 are used in the first DES operation, the second 64 bits in the second (middle) DES operation, and the
894 third 64 bits in the third (last) DES operation. Each of these 64 bits of key contain 56 effective bits and
895 8 parity bits.

896

897 The algorithm standard is denoted by the URL: <http://www.w3.org/2001/04/xmlenc#tripledes-cbc>.

898 2. Advanced Encryption Standard (AES) – 128-bit

899

900 Within the context of this specification, and as specified in [AES], this is a 128-bit symmetric key used in
901 the Cipher Block Chaining (CBC) mode.

902

903 The algorithm standard is denoted by the URL: <http://www.w3.org/2001/04/xmlenc#aes128-cbc>.

904 3. Advanced Encryption Standard (AES) – 192-bit
905
906 Within the context of this specification, and as specified in [AES], this is a 192-bit symmetric key used in
907 the Cipher Block Chaining (CBC) mode.

908
909 The algorithm standard is denoted by the URL: <http://www.w3.org/2001/04/xmlenc#aes192-cbc>.

910 4. Advanced Encryption Standard (AES) – 256-bit

911
912 Within the context of this specification, and as specified in [AES], this is a 256-bit symmetric key used in
913 the Cipher Block Chaining (CBC) mode.

914
915 The algorithm standard is denoted by the URL: <http://www.w3.org/2001/04/xmlenc#aes256-cbc>.

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

917 Some examples of the <KeyAlgorithm> element are as follows; other elements of the <KeyUsePolicy> element
918 are not displayed for brevity:

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

```
920 <ekmi:KeyUsePolicy>
921   ...
922   <ekmi:KeyAlgorithm>
923     http://www.w3.org/2001/04/xmlenc#tripleDES-cbc
924   </ekmi:KeyAlgorithm>
925   ...
926 </ekmi:KeyUsePolicy>
```

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

```
928 <ekmi:KeyUsePolicy>
929   ...
930   <ekmi:KeyAlgorithm>
931     http://www.w3.org/2001/04/xmlenc#aes128-cbc
932   </ekmi:KeyAlgorithm>
933   ...
934 </ekmi:KeyUsePolicy>
```

935 2.10 Element <KeySize>

936 The element <KeySize>, of type *KeySizeType*, is used to identify the size of the symmetric key, in binary digits
937 (bits) in the <SymkeyResponse>.

938 **Schema Definition:**

```
939 <xsd:simpleType name="KeySizeType">
940   <xsd:restriction base="xsd:unsignedShort">
941     <xsd:totalDigits value="3"/>
942     <xsd:fractionDigits value="0"/>
943     <xsd:enumeration value="128"/>
944     <xsd:enumeration value="192"/>
945     <xsd:enumeration value="256"/>
946   </xsd:restriction>
947 </xsd:simpleType>
```

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

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

950 1. 128-bits when used with the *AES-192* algorithm

- 951 2. 192-bits when used with the *AES-192* or the *3DES* algorithms
- 952 3. 256-bits when used with the *AES-256* algorithm

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

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

```
956 <ekmi:KeyUsePolicy>  
957   ...  
958   <ekmi:KeySize>128</ekmi:KeySize>  
959   ...  
960 </ekmi:KeyUsePolicy>
```

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

```
962 <ekmi:KeyUsePolicy>  
963   ...  
964   <ekmi:KeySize>192</ekmi:KeySize>  
965   ...  
966 </ekmi:KeyUsePolicy>
```

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

```
968 <ekmi:KeyUsePolicy>  
969   ...  
970   <ekmi:KeySize>256</ekmi:KeySize>  
971   ...  
972 </ekmi:KeyUsePolicy>
```

973 2.11 Element <Status>

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

976 **Schema Definition:**

```
977 <xsd:simpleType name="StatusType">  
978   <xsd:restriction base="xsd:string">  
979     <xsd:enumeration value="Active"/>  
980     <xsd:enumeration value="Default"/>  
981     <xsd:enumeration value="Inactive"/>  
982     <xsd:enumeration value="Other"/>  
983   </xsd:restriction>  
984 </xsd:simpleType>
```

985 Where it does exist, there SHALL be only one <Status> element within the enclosing element.

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

- 987 1. The **Active** value indicates that the element that makes up the document-root is currently active in the
988 SKMS and conforming **SKCL** implementations may use it within applications.
- 989 2. The **Default** value indicates that the element that makes up the document root is the default element in
990 the SKMS, is also active, and conforming **SKCL** implementations may use it within applications.
- 991 3. The **Inactive** value indicates that the element that makes up the document root is not active in the
992 SKMS, and conforming **SKCL** implementations may NOT use it within applications.
- 993 4. The **Other** value indicates that the element that makes up the document root has a meaning that is
994 application-specific. However, conforming **SKCL** implementations may NOT use it within applications.

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

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

```
998     <ekmi:KeyUsePolicy>  
999         ...  
1000         <ekmi:Status>Active</ekmi:Status>  
1001         ...  
1002     </ekmi:KeyUsePolicy>
```

1003 **Example 2 – An example with an Inactive status within a <KeyUsePolicy> element:**

```
1004     <ekmi:KeyUsePolicy>  
1005         ...  
1006         <ekmi:Status>Inactive</ekmi:Status>  
1007         ...  
1008     </ekmi:KeyUsePolicy>
```

1009 **Example 3 – An example with a Default status within a <KeyUsePolicy> element:**

```
1010     <ekmi:KeyUsePolicy>  
1011         ...  
1012         <ekmi:Status>Default</ekmi:Status>  
1013         ...  
1014     </ekmi:KeyUsePolicy>
```

1015 2.12 Element <Permissions>

1016 The <Permissions> element, of the type *PermissionsType* is at the heart of the <KeyUsePolicy> element. It
1017 provides guidance to conforming SKCL implementations on who may use the symmetric key, when they may use
1018 it, for what purposes, for how long and in which locations. For applications that conform to the Multi-Level
1019 Security (MLS) model, there is a provision for specifying which levels are permitted use of the key. There is also
1020 an element that allows for extending the <Permissions> element to accommodate rules that have not been
1021 envisioned in the current specification.

1022 There SHALL be only one <Permissions> element within a <KeyUsePolicy> element.

1023 **Schema Definition:**

```
1024     <xsd:complexType name="PermissionsType">  
1025         <xsd:sequence>  
1026             <xsd:element  
1027                 name="PermittedApplications"  
1028                 type="tns:PermittedApplicationsType"  
1029                 minOccurs="0"/>  
1030             <xsd:element  
1031                 name="PermittedDates"  
1032                 type="tns:PermittedDatesType"  
1033                 minOccurs="0"/>  
1034             <xsd:element  
1035                 name="PermittedDays"  
1036                 type="tns:PermittedDaysType"  
1037                 minOccurs="0"/>  
1038             <xsd:element  
1039                 name="PermittedDuration"  
1040                 type="tns:PermittedDurationType"  
1041                 minOccurs="0"/>  
1042             <xsd:element  
1043                 name="PermittedLevels"  
1044                 type="tns:PermittedLevelsType"
```

```

1045         minOccurs="0"/>
1046     <xsd:element
1047         name="PermittedLocations"
1048         type="tns:PermittedLocationsType"
1049         minOccurs="0"/>
1050     <xsd:element
1051         name="PermittedNumberOfTransactions"
1052         type="tns:PermittedNumberOfTransactionsType"
1053         minOccurs="0"/>
1054     <xsd:element
1055         name="PermittedTimes"
1056         type="tns:PermittedTimesType"
1057         minOccurs="0"/>
1058     <xsd:element
1059         name="PermittedUses"
1060         type="tns:PermittedUsesType"
1061         minOccurs="0"/>
1062     <xsd:element
1063         name="Other"
1064         type="xsd:anyType"
1065         minOccurs="0"/>
1066 </xsd:sequence>
1067 </xsd:complexType>

```

1068
1069 The <Permissions> element consists of the following sub-elements:

- 1070 1. The optional <PermittedApplications> element identifies applications that are permitted the use of the
1071 symmetric key in question.
1072
1073 The absence of the <PermittedApplications> element in a <Permissions> element implies that all
1074 applications are permitted to use the key. Identifying a specific application restricts the use of the key to
1075 only the identified applications.
1076
1077 The <PermittedApplications> element is specified in Section 2.13.
- 1078 2. The optional <PermittedDates> element identifies the dates during which applications are permitted the
1079 use of the symmetric key in question.
1080
1081 The absence of the <PermittedDates> element in a <Permissions> element implies that applications are
1082 permitted to use the key on any date. Identifying specific dates restricts the use of the key to only the
1083 duration between the identified dates.
1084
1085 The <PermittedDates> element is specified in Section 2.12.
- 1086 3. The optional <PermittedDays> element identifies the days of week during which applications are
1087 permitted the use of the symmetric key in question.
1088
1089 The absence of the <PermittedDays> element in a <Permissions> element implies that applications are
1090 permitted to use the key on any day of the week. Identifying specific days restricts the use of the key to
1091 only the identified days.
1092
1093 The <PermittedDays> element is specified in Section 2.15.
- 1094 4. The optional <PermittedDuration> element identifies the duration (in seconds) in which applications are
1095 permitted the use of the symmetric key in question, once the SKCL starts using the symmetric key.
1096
1097 The absence of the <PermittedDuration> element in a <Permissions> element implies that applications
1098 are permitted to use the key for any duration after it has been used. Identifying the specific duration
1099 restricts the use of the key to only the period after the start of the use of the key.
1100

1101 A distinction between <PermittedDates> and <PermittedDuration> is that the former has fixed start and
1102 end-dates for the use of the key, whereas the latter has a fixed end-date-and-time after the key has
1103 begun to be used without a fixed start-date-and-time. Thus, an application with a <PermittedDuration>
1104 can begin the use of a symmetric key at any time, but must stop its use at the end of the
1105 <PermittedDuration> once it has begun. With <PermittedDates>, an application can continue using the
1106 symmetric key until the fixed date-and-time have been reached.

1107
1108 The <PermittedDuration> element is specified in Section 2.x16

- 1109 5. Within a Multi-Level Security (MLS) system, the optional <PermittedLevels> element identifies the levels
1110 at which applications are permitted the use of the symmetric key in question.

1111
1112 The absence of the <PermittedLevels> element in a <Permissions> element implies that applications are
1113 permitted to use the key on any level of security. Identifying specific level(s) restricts the use of the key
1114 to only the identified level(s).

1115
1116 The <PermittedLevels> element is specified in Section 2.x17

- 1117 6. The optional <PermittedLocations> element identifies physical geographic locations where applications
1118 are permitted the use of the symmetric key in question.

1119
1120 The absence of the <PermittedLocations> element in a <Permissions> element implies that applications
1121 are permitted to use the key at any physical location. Identifying specific locations restricts the use of
1122 the key to only the identified locations.

1123
1124 The <PermittedLocations> element is specified in Section 2.18.

- 1125 7. The optional <PermittedNumberOfTransactions> element identifies the number of encryption
1126 transactions that applications are permitted, with the use of the symmetric key in question.

1127
1128 The absence of the <PermittedNumberOfTransactions> element in a <Permissions> element implies
1129 that applications are permitted to use the key for as many encryption transactions as necessary.
1130 Identifying a specific number of transactions restricts the use of the key to only the limit identified in the
1131 element.

1132
1133 The <PermittedNumberOfTransactions> element is specified in Section 2.19.

- 1134 8. The optional <PermittedTimes> element identifies the times of day during which applications are
1135 permitted the use of the symmetric key in question.

1136
1137 The absence of the <PermittedTimes> element in a <Permissions> element implies that applications are
1138 permitted to use the key at any time of day. Identifying specific times restricts the use of the key to only
1139 the duration of the identified times.

1140
1141 The <PermittedTimes> element is specified in Section 2.20.

- 1142 9. The optional <PermittedUses> element identifies the uses which applications are permitted with the
1143 symmetric key in question.

1144
1145 The absence of the <PermittedUses> element in a <Permissions> element implies that applications are
1146 permitted to use the key for any purpose. Identifying specific uses restricts the use of the key to only
1147 the identified uses.

1148
1149 The <PermittedUses> element is specified in Section 2.21.

- 1150 10. The optional <Other> element allows implementers to specify permissions that cannot be addressed
1151 with the above-mentioned categories, for restricting the use of the symmetric key in question.

1152
1153 While the <Other> element provides flexibility for implementations, the disadvantage of the element is
1154 that it may render a specific implementation incompatible with the **SKSML** standard. It is strongly
1155 recommended that implementers avoid the use of the <Other> element unless they definitely do not

1156 expect to inter-operate with other **SKCL** implementations. If there is a strong need for capability that
1157 does not exist within the current <Permissions> element, implementers are encouraged to contact the
1158 OASIS EKMI TC and raise their concerns.

1159 When the <Permissions> element is empty, that there are no restrictions on the use of the symmetric key other
1160 than that the application calling on the **SKCL** be authorized to access the key in question. However, when there
1161 are elements defined within the <Permissions> element, conforming **SKCL** implementations must comply with all
1162 the permission elements, evaluating the most restrictive permissions first and in decreasing order of restriction,
1163 before allowing the use of the key.

1164 For example, if a <Permissions> element specifies that a key may be used on Weekdays, between the hours of
1165 0900 and 1700 Hours, then a request for a symmetric key on a Saturday at 1105 would deny use of the key in
1166 question, since it violates the more restrictive permission of being allowed for use only on weekdays. It should be
1167 noted that the **SKS** server will generate the key – or return an existing key - when an authorized client with
1168 appropriate access requests it. However, it is up to the **SKCL** implementation to comply with the rules in the
1169 <Permissions> element.

1170 In another example, if a <Permissions> element specifies a <PermittedDuration> of 600 seconds from the start of
1171 use of the key, and a <PermittedNumberOfTransactions> of 10 encryption transactions, conforming **SKCL**
1172 implementations must evaluate both permissions before each transaction and determine if they are both within
1173 the specified thresholds before using the key. If the 600 seconds expire before the 10 encryption transactions
1174 have been completed, or if the 10 encryption transactions are completed before 600 seconds have expired,
1175 conforming **SKCL** implementations **MUST** not use the key in question anymore.

1176 Some examples of the <Permissions> element are as follows; the enclosing <KeyUsePolicy> element, <Symkey>
1177 element and <SymkeyResponse> elements are not displayed for brevity:

1178 **Example 1 – a collection of permissions that permits a single application the use of the symmetric key in**
1179 **question, between January 01, 2008 and December 31, 2008 and between the hours of 0700 and 1900.**
1180 **(Day Of Week restrictions are discussed in Section 2.15):**

```
1181 <ekmi:Permissions>
1182   <ekmi:PermittedApplications>
1183     <ekmi:PermittedApplication>
1184       <ekmi:ApplicationID>10514-23</ekmi:ApplicationID>
1185       <ekmi:ApplicationName>Payroll Application</ekmi:ApplicationName>
1186       <ekmi:Version>1.0</ekmi:Version>
1187       <ekmi:DigestAlgorithm>http://www.w3.org/2000/09/xmlsig#sha1</ekmi:DigestAlgorithm>
1188       <ekmi:DigestValue>NIG4bKkt4cziEqFFuOoBTM81efU=</ekmi:DigestValue>
1189     </ekmi:PermittedApplication>
1190   </ekmi:PermittedApplications>
1191   <ekmi:PermittedDates>
1192     <ekmi:PermittedDate>
1193       <ekmi:StartDate>2008-01-01</ekmi:StartDate>
1194       <ekmi:EndDate>2008-12-31</ekmi:EndDate>
1195     </ekmi:PermittedDate>
1196   </ekmi:PermittedDates>
1197   <ekmi:PermittedTimes>
1198     <ekmi:PermittedTime>
1199       <ekmi:StartTime>07:00:00</ekmi:StartTime>
1200       <ekmi:EndTime>19:00:00</ekmi:EndTime>
1201     </ekmi:PermittedTime>
1202   </ekmi:PermittedTimes>
1203 </ekmi:Permissions>
```

1204 **Example 2 – a collection of permissions that permits two specific applications the use of the symmetric**
1205 **key in question, between January 01, 2009 and January 31, 2009.**

```
1206 <ekmi:Permissions>
1207   <ekmi:PermittedApplications>
1208     <ekmi:PermittedApplication>
1209       <ekmi:ApplicationID>10514-24</ekmi:ApplicationID>
```

```

1210     <ekmi:ApplicationName>Employee Tax Reporting Application</ekmi:ApplicationName>
1211     <ekmi:Version>3.3</ekmi:Version>
1212     <ekmi:DigestAlgorithm>http://www.w3.org/2000/09/xmldsig#sha1</ekmi:DigestAlgorithm>
1213     <ekmi:DigestValue>G4bsdfKkt4cziEqFFuOoBTM81efU=</ekmi:DigestValue>
1214   </ekmi:PermittedApplication>
1215   <ekmi:PermittedApplication>
1216     <ekmi:ApplicationID>10514-25</ekmi:ApplicationID>
1217     <ekmi:ApplicationName>IRS Tax Reporting Application</ekmi:ApplicationName>
1218     <ekmi:Version>2.1</ekmi:Version>
1219     <ekmi:DigestAlgorithm>http://www.w3.org/2000/09/xmldsig#sha1</ekmi:DigestAlgorithm>
1220     <ekmi:DigestValue>4uyb4sd234cziqzlmnuOoBTMa08=</ekmi:DigestValue>
1221   </ekmi:PermittedApplication>
1222 </ekmi:PermittedApplications>
1223 <ekmi:PermittedDates>
1224   <ekmi:PermittedDate>
1225     <ekmi:StartDate>2009-01-01</ekmi:StartDate>
1226     <ekmi:EndDate>2009-01-31</ekmi:EndDate>
1227   </ekmi:PermittedDate>
1228 </ekmi:PermittedDates>
1229 </ekmi:Permissions>

```

1230 **Example 3 – a collection of permissions that permits all applications the use of the symmetric key in**
1231 **question, for 100 transactions for encrypting/decrypting credit card numbers.**

```

1232   <ekmi:Permissions>
1233     <ekmi:PermittedNumberOfTransactions>100</ekmi:PermittedNumberOfTransactions>
1234     <ekmi:PermittedUses>
1235       <ekmi:PermittedUse>CCN</ekmi:PermittedUse>
1236     </ekmi:PermittedUses>
1237   </ekmi:Permissions>

```

1238 **Example 4 – a collection of permissions that permits all applications the use of the symmetric key in**
1239 **question, for 600 seconds once the SKCL starts using the key.**

```

1240   <ekmi:Permissions>
1241     <ekmi:PermittedDuration>600</ekmi:PermittedDuration>
1242   </ekmi:Permissions>

```

1243 **Example 5 – a collection of permissions that permits a specific application the use of the symmetric key**
1244 **in question, at specific geographic locations on weekdays between the hours of 0800 and 1700, and only**
1245 **when the application is operating at the Secret level in an MLS system.**

```

1246   <ekmi:Permissions>
1247     <ekmi:PermittedDays>
1248       <ekmi:PermittedDay>Weekday</ekmi:PermittedDay>
1249     </ekmi:PermittedDays>
1250     <ekmi:PermittedLocations>
1251       <ekmi:PermittedLocation>
1252         <ekmi:LocationName>Facility A51</ekmi:LocationName>
1253         <ekmi:Latitude>37.385562</ekmi:Latitude>
1254         <ekmi:Longitude>-121.993387</ekmi:Latitude>
1255       </ekmi:PermittedLocation>
1256       <ekmi:PermittedLocation>
1257         <ekmi:LocationName>Facility DC-VA01</ekmi:LocationName>
1258         <ekmi:Latitude>88.485362</ekmi:Latitude>
1259         <ekmi:Longitude>-21.453648</ekmi:Latitude>
1260       </ekmi:PermittedLocation>
1261     </ekmi:PermittedLocations>
1262     <ekmi:PermittedLevels>
1263       <ekmi:PermittedLevel>Secret</ekmi:PermittedLevel>
1264     </ekmi:PermittedLevels>

```

```

1265     <ekmi:PermittedTimes>
1266         <ekmi:PermittedTime>
1267             <ekmi:StartTime>07:00:00</ekmi:StartTime>
1268             <ekmi:EndTime>19:00:00</ekmi:EndTime>
1269         </ekmi:PermittedTime>
1270     </ekmi:PermittedTimes>
1271 </ekmi:Permissions>

```

1272 2.13 Element <PermittedApplications> and <PermittedApplication>

1273 The element <PermittedApplications> , of type *PermittedApplicationsType* and its only child-element
1274 <PermittedApplication> of type *ApplicationsType* are used to define the list of applications that are permitted to
1275 use a symmetric key within a specific <Symkey> element.

1276 Schema Definition:

```

1277 <xsd:complexType name="PermittedApplicationsType">
1278     <xsd:sequence>
1279         <xsd:element
1280             name="PermittedApplication"
1281             type="tns:ApplicationsType"
1282             minOccurs="0"
1283             maxOccurs="unbounded"/>
1284     </xsd:sequence>
1285 </xsd:complexType>

```

1286 Schema Definition:

```

1287 <xsd:complexType name="ApplicationsType">
1288     <xsd:sequence>
1289         <xsd:element name="ApplicationID" type="tns:TwoPartIDType"/>
1290         <xsd:element name="ApplicationName">
1291             <xsd:simpleType>
1292                 <xsd:restriction base="xsd:string">
1293                     <xsd:maxLength value="256"/>
1294                     <xsd:whiteSpace value="preserve"/>
1295                 </xsd:restriction>
1296             </xsd:simpleType>
1297         </xsd:element>
1298         <xsd:element name="Version" minOccurs="0">
1299             <xsd:simpleType>
1300                 <xsd:restriction base="xsd:string">
1301                     <xsd:maxLength value="32"/>
1302                     <xsd:whiteSpace value="preserve"/>
1303                 </xsd:restriction>
1304             </xsd:simpleType>
1305         </xsd:element>
1306         <xsd:group ref="tns:MessageDigestGroup" minOccurs="0"/>
1307         <xsd:element name="Other" type="xsd:anyType" minOccurs="0"/>
1308     </xsd:sequence>
1309 </xsd:complexType>

```

1311 Schema Definition:

```

1312 <xsd:group name="MessageDigestGroup">
1313     <xsd:sequence>
1314         <xsd:element name="DigestAlgorithm">
1315             <xsd:simpleType>
1316                 <xsd:restriction base="xsd:anyURI">
1317                     <xsd:enumeration value="http://www.w3.org/2000/09/xmlsig#sha1"/>
1318                 </xsd:restriction>

```

```

1319         <xsd:enumeration value="http://www.w3.org/2001/04/xmlenc#sha256"/>
1320         <xsd:enumeration value="http://www.w3.org/2001/04/xmlenc#sha512"/>
1321     </xsd:restriction>
1322 </xsd:simpleType>
1323 </xsd:element>
1324 <xsd:element name="DigestValue">
1325     <xsd:simpleType>
1326         <xsd:restriction base="xsd:base64Binary">
1327             <xsd:maxLength value="1024"/>
1328         </xsd:restriction>
1329     </xsd:simpleType>
1330 </xsd:element>
1331 </xsd:sequence>
1332 </xsd:group>

```

1333 When it is used within a <KeyUsePolicy> element, there SHALL be only one <PermittedApplications> element
1334 within the <KeyUsePolicy> element. However, there MAY be an unbounded (unlimited) number of
1335 <PermittedApplication> elements within a <PermittedApplications> element.

1336 NOTE: It is noteworthy to mention that the absence of a <PermittedApplications> element within a
1337 <KeyUsePolicy> element specifies that ALL applications are permitted the use of the symmetric key, subject to
1338 complying with all other permission clauses in the <KeyUsePolicy> element, if any.

1339 The <PermittedApplication> element provides details of the application that is permitted to use the symmetric
1340 key in question. The <PermittedApplication> element consists of the following sub-elements:

- 1341 1. The <ApplicationID> element identifies the unique identifier assigned to this application within the
1342 SKMS. It is a **TwoPartIDType** as specified in Section 2.8. There SHALL be only one <ApplicationID>
1343 element within a <PermittedApplication> element.
 - 1344 2. The <ApplicationName> element identifies the name assigned to this application within the SKMS. It is
1345 an XSD **String** with a maximum length of 256 characters. There SHALL be only one
1346 <ApplicationName> element within a <PermittedApplication> element.
 - 1347 3. An optional <Version> element identifying the version number of this application within the SKMS. It is
1348 an XSD **String** with a maximum length of 32 characters. There MAY be only one <Version> element
1349 within a <PermittedApplication> element.
 - 1350 4. The <MessageDigestGroup> group which identifies the message digest of the application's binary
1351 image, along with the message digest algorithm used to calculate the digest value. The
1352 <MessageDigestGroup> consists of the following elements:
 - 1353 a) The <DigestAlgorithm> element of the XSD type **anyURI**, which supports one of the following
1354 three digest algorithms:
 - 1355 i. <http://www.w3.org/2000/09/xmldsig#sha1>
 - 1356 ii. <http://www.w3.org/2001/04/xmlenc#sha256>
 - 1357 iii. <http://www.w3.org/2001/04/xmlenc#sha512>
 - 1358 b) The <DigestValue> element of the XSD type **base64Binary**.
- 1359 There SHALL be only one <MessageDigestGroup> group within a <PermittedApplication> element.
- 1360 5. An optional <Other> element that provides implementers the ability to carry other information about the
1361 application that may be relevant to their SKMS. Implementers are cautioned that the use of the
1362 <Other> element may not be supported by other SKCL implementations, and may break interoperability
1363 of the SKSML protocol. Should there be a strong need for additional features in the
1364 <PermittedApplication> element, implementers are encouraged to contact the OASIS EKMI TC with their
1365 requirements.

1366 NOTE: The **SKSML** specification does not specify how an **SKCL** implementation will determine the message
1367 digest of application that needs to use the symmetric key in question. It is left to the implementers of the **SKCL**
1368 how they determine the message digest of the calling application, with the specified algorithm, before it permits
1369 the application the use of the symmetric key.

1370 Some examples of the <PermittedApplications> element are shown below; other parts of their enclosing
1371 elements are not shown for brevity:

1372 **Example 1 – An example of a <PermittedApplications> element with two child <PermittedApplication>**
1373 **elements with specific version numbers and message digest values:**

```
1374 <ekmi:PermittedApplications>
1375   <ekmi:PermittedApplication>
1376     <ekmi:ApplicationID>10514-24</ekmi:ApplicationID>
1377     <ekmi:ApplicationName>Employee Tax Reporting</ekmi:ApplicationName>
1378     <ekmi:Version>3.3</ekmi:Version>
1379     <ekmi:DigestAlgorithm>
1380       http://www.w3.org/2000/09/xmlsig#sha1
1381     </ekmi:DigestAlgorithm>
1382     <ekmi:DigestValue>G4bsdfKkt4cziEqFFu0oBTM81efU=</ekmi:DigestValue>
1383   </ekmi:PermittedApplication>
1384   <ekmi:PermittedApplication>
1385     <ekmi:ApplicationID>10514-25</ekmi:ApplicationID>
1386     <ekmi:ApplicationName>IRS Tax Reporting Application</ekmi:ApplicationName>
1387     <ekmi:Version>2.1</ekmi:Version>
1388     <ekmi:DigestAlgorithm>
1389       http://www.w3.org/2001/04/xmlenc#sha256
1390     </ekmi:DigestAlgorithm>
1391     <ekmi:DigestValue>
1392       ab7b85c9410d48c54fc7939c391be4028e7305085191c56e7b3740f2cbdbbc79
1393     </ekmi:DigestValue>
1394   </ekmi:PermittedApplication>
1395 </ekmi:PermittedApplications>
```

1396 **Example 1 – An example of a <PermittedApplications> element with one child <PermittedApplication>**
1397 **element that applies to all versions of the application; the message digest value and algorithm are not**
1398 **used in this example:**

```
1399 <ekmi:PermittedApplications>
1400   <ekmi:PermittedApplication>
1401     <ekmi:ApplicationID>10514-14</ekmi:ApplicationID>
1402     <ekmi:ApplicationName>E-Commerce Payment</ekmi:ApplicationName>
1403   </ekmi:PermittedApplication>
1404 </ekmi:PermittedApplications>
```

1405 **2.14 Element <PermittedDates> and <PermittedDate>**

1406 The element <PermittedDates> , of type **PermittedDatesType** and its only child-element <PermittedDate> ,
1407 which is an anonymous XSD **ComplexType**, are used to define sets of dates between which applications are
1408 permitted to use a symmetric key within a specific <Symkey> element.

1409 **Schema Definition:**

```
1410 <xsd:complexType name="PermittedDatesType">
1411   <xsd:sequence>
1412     <xsd:element name="PermittedDate" minOccurs="0" maxOccurs="unbounded">
1413       <xsd:complexType>
1414         <xsd:sequence>
1415           <xsd:element name="StartDate">
1416             <xsd:simpleType>
1417               <xsd:restriction base="xsd:date">
```

```

1418         <xsd:pattern value="\p{Nd}{4}-\p{Nd}{2}-\p{Nd}{2}"/>
1419         </xsd:restriction>
1420     </xsd:simpleType>
1421 </xsd:element>
1422 <xsd:element name="EndDate">
1423     <xsd:simpleType>
1424         <xsd:restriction base="xsd:date">
1425             <xsd:pattern value="\p{Nd}{4}-\p{Nd}{2}-\p{Nd}{2}"/>
1426         </xsd:restriction>
1427     </xsd:simpleType>
1428 </xsd:element>
1429 </xsd:sequence>
1430 </xsd:complexType>
1431 </xsd:element>
1432 </xsd:sequence>
1433 </xsd:complexType>

```

1434 When it is used within a <KeyUsePolicy> element, there SHALL be only one <PermittedDates> element within
1435 the <KeyUsePolicy> element. However, there MAY be an unbounded number of <PermittedDate> elements
1436 within a <PermittedDates> element.

1437 NOTE: It is noteworthy to mention that the absence of a <PermittedDates> element within a <KeyUsePolicy>
1438 element specifies that applications are permitted the use of the symmetric key on any calendar date of the year,
1439 subject to complying with all other permission clauses in the <KeyUsePolicy> element, if any.

1440 The <PermittedDate> element identifies an individual set of dates between which application are permitted to use
1441 the symmetric key in question. The <PermittedDate> element consists of the following sub-elements:

- 1442 1. The <StartDate> element identifies the date from which applications may start using the symmetric key
1443 in question. It is an XSD **Date** type that MUST be specified in a specific pattern (see examples) where
1444 the first four digits specify the year, the second two digits specify the calendar month in the year, and
1445 the last two digits specify the calendar date of the month.

1446 There SHALL be only one <StartDate> element within a <PermittedDate> element.

1447 Conforming **SKCL** implementations SHALL NOT start using the symmetric before the onset of the
1448 <StartDate> on the client machine. Unless further constrained by the <PermittedTimes> element, the
1449 onset of the <StartDate> is specified to be the first second of the day – 00:00:01 Hours – on the client
1450 machine.
1451
1452

- 1453 2. The <EndDate> element identifies the date until which applications may use the symmetric key in
1454 question. It is an XSD **Date** type that MUST be specified in a specific pattern (see examples) where the
1455 first four digits specify the year, the second two digits specify the calendar month in the year, and the
1456 last two digits specify the calendar date of the month.

1457 There SHALL be only one <EndDate> element within a <PermittedDate> element.

1458 Conforming **SKCL** implementations SHALL NOT use the symmetric after the end of the <EndDate> on
1459 the client machine. Unless further constrained by the <PermittedTimes> element, the end of the
1460 <EndDate> is specified to be the last second of the day – 23:59:59 Hours – on the client machine.
1461
1462

1463 Some examples of the <PermittedDates> element are shown below; other required parts of their enclosing
1464 elements are not shown for brevity:

1465 **Example 1 – An example of a <PermittedDates> element with a single <PermittedDate> element. The**
1466 **<StartDate> specifies January 01, 2009 while the <EndDate> specifies January 31, 2009:**

```

1467 <ekmi:PermittedDates>
1468     <ekmi:PermittedDate>
1469         <ekmi:StartDate>2009-01-01</ekmi:StartDate>
1470         <ekmi:EndDate>2009-01-31</ekmi:EndDate>

```

```
1471     </ekmi:PermittedDate>
1472 </ekmi:PermittedDates>
```

1473 **Example 2 – An example of a <PermittedDates> element with two <PermittedDate> elements. For the first**
1474 **<PermittedDate> element , the <StartDate> element specifies July 01, 2008 while the <EndDate> element**
1475 **specifies July 03, 2008. For the second <PermittedDate> element, the <StartDate> element specifies July**
1476 **07, 2008 while the <EndDate> element specifies July 12, 2008. This policy would restrict a symmetric key**
1477 **with this <PermittedDates> element so it cannot be used on the July 4th weekend of 2008:**

```
1478     <ekmi:PermittedDates>
1479         <ekmi:PermittedDate>
1480             <ekmi:StartDate>2008-07-01</ekmi:StartDate>
1481             <ekmi:EndDate>2008-07-03</ekmi:EndDate>
1482         </ekmi:PermittedDate>
1483         <ekmi:PermittedDate>
1484             <ekmi:StartDate>2008-07-07</ekmi:StartDate>
1485             <ekmi:EndDate>2009-07-12</ekmi:EndDate>
1486         </ekmi:PermittedDate>
1487     </ekmi:PermittedDates>
```

1488 2.15 Element <PermittedDays> and <PermittedDay>

1489 The element <PermittedDays> , of the type *PermittedDaysType* and its only child-element <PermittedDay> of
1490 the *PermittedDayType*, are used to define days of the week when applications are permitted to use a
1491 symmetric key within a specific <Symkey> element.

1492 Schema Definition:

```
1493     <xsd:complexType name="PermittedDaysType">
1494         <xsd:sequence>
1495             <xsd:element
1496                 name="PermittedDay"
1497                 type="tns:PermittedDayType"
1498                 minOccurs="0"
1499                 maxOccurs="unbounded">
1500             </xsd:element>
1501         </xsd:sequence>
1502     </xsd:complexType>
```

1503 Schema Definition:

```
1504     <xsd:simpleType name="PermittedDayType">
1505         <xsd:restriction base="xsd:string">
1506             <xsd:enumeration value="Sunday"/>
1507             <xsd:enumeration value="Monday"/>
1508             <xsd:enumeration value="Tuesday"/>
1509             <xsd:enumeration value="Wednesday"/>
1510             <xsd:enumeration value="Thursday"/>
1511             <xsd:enumeration value="Friday"/>
1512             <xsd:enumeration value="Saturday"/>
1513             <xsd:enumeration value="Weekday"/>
1514             <xsd:enumeration value="Weekend"/>
1515         </xsd:restriction>
1516     </xsd:simpleType>
```

1517 When it is used within a <KeyUsePolicy> element, there SHALL be only one <PermittedDay> element within the
1518 <KeyUsePolicy> element. However, there MAY be an unbounded (unlimited) number of <PermittedDay>
1519 elements within a <PermittedDays> element.

1520 NOTE: It is noteworthy to mention that the absence of a <PermittedDays> element within a <KeyUsePolicy>
1521 element specifies that applications are permitted the use of the symmetric key on all days of the week, subject to
1522 complying with all other permission clauses in the <KeyUsePolicy> element, if any.

1523 The <PermittedDay> element, of the XSD *String* type, identifies individual days of the week – from an
1524 enumerated list - on which application are permitted to use the symmetric key in question.

1525 Some examples of the <PermittedDays> element are shown below; other parts of their enclosing elements are
1526 not shown for brevity:

1527 **Example 1 – An example of a <PermittedDays> element with a single<PermittedDay> child element,**
1528 **specifying that the symmetric key may be used only on weekdays:**

```
1529     <ekmi:PermittedDays>  
1530         <ekmi:PermittedDay>Weekday</ekmi:PermittedDay>  
1531     </ekmi:PermittedDays>
```

1532 **Example 2 – An example of a <PermittedDays> element with three <PermittedDay> child elements,**
1533 **specifying that the symmetric key may be used only on Mondays, Wednesdays and Fridays:**

```
1534     <ekmi:PermittedDays>  
1535         <ekmi:PermittedDay>Monday</ekmi:PermittedDay>  
1536         <ekmi:PermittedDay>Wednesday</ekmi:PermittedDay>  
1537         <ekmi:PermittedDay>Friday</ekmi:PermittedDay>  
1538     </ekmi:PermittedDays>
```

1539 2.16 Element <PermittedDuration>

1540 The element <PermittedDuration> , of the type *PermittedDurationType* is used to define the number of
1541 seconds during which applications are permitted to use a symmetric key within a specific <Symkey> element,
1542 once the SKCL has started using the symmetric key in question.

1543 Schema Definition:

```
1544     <xsd:simpleType name="PermittedDurationType">  
1545         <xsd:restriction base="xsd:positiveInteger">  
1546             <xsd:maxInclusive value="18446744073709551615"/>  
1547         </xsd:restriction>  
1548     </xsd:simpleType>
```

1549 When it is used within a <KeyUsePolicy> element, there SHALL be only one <PermittedDuration> element within
1550 the <KeyUsePolicy> element.

1551 NOTE: It is noteworthy to mention that the absence of a <PermittedDuration> element within a <KeyUsePolicy>
1552 element specifies that applications are permitted the use of the symmetric key forever, subject to complying with
1553 all other permission clauses in the <KeyUsePolicy> element, if any.

1554 The <PermittedDuration> element, of the XSD *positiveInteger* type, identifies the precise number of seconds for
1555 which the symmetric key in question may be used ONCE the key has been used by conforming SKCL
1556 implementations for the first time.

1557 However, as long as the symmetric has not been used by an SKCL on a client device – it might be cached for
1558 many days/weeks/months depending on the <KeyCachePolicy> in effect within the SKMS for that device – the
1559 effective lifetime of the symmetric key may be well past the number of seconds specified in <PermittedDuration>
1560 when calculated from the time of the key's generation time on the SKS server. It is the responsibility of the SKCL
1561 implementation , when presented with a <PermittedDuration> element in a <KeyUsePolicy> of a symmetric key,
1562 to keep track of the date/time when the symmetric key in question was first used on the client device, and how
1563 long the key will last after that.

1564 Some examples of the <PermittedDuration> element are shown below; other parts of their enclosing elements
1565 are not shown for brevity:

1566 **Example 1 – An example of a <PermittedDuration> element specifying that the symmetric key may be**
1567 **used only for a single 24-hour period from the moment it is first used by an SKCL:**

```
1568 <ekmi:PermittedDuration>86400</ekmi:PermittedDuration>
```

1569 **Example 2 – An example of a <PermittedDuration> element specifying that the symmetric key may be**
1570 **used only for week from the moment it is first used by an SKCL:**

```
1571 <ekmi:PermittedDuration>604800</ekmi:PermittedDuration>
```

1572 **Example 3 – An example of a <PermittedDuration> element specifying that the symmetric key may be**
1573 **used only 5 minutes from the moment it is first used by an SKCL:**

```
1574 <ekmi:PermittedDuration>300</ekmi:PermittedDuration>
```

1575 **2.17 Element <PermittedLevels> and <PermittedLevel>**

1576 The element <PermittedLevels>, of the type *LevelClassificationType*, is used to define the security level at
1577 which applications are permitted to use a symmetric key within a specific <Symkey> element. This element is
1578 useful only to applications and systems that conform to the multi-level security (MLS) system as defined in the
1579 Bell-LaPadula model.

1580 **Schema Definition:**

```
1581 <xsd:complexType name="PermittedLevelsType">  
1582 <xsd:sequence>  
1583 <xsd:element  
1584 name="PermittedLevel"  
1585 type="tns:LevelClassificationType"  
1586 minOccurs="0"  
1587 maxOccurs="unbounded">  
1588 </xsd:element>  
1589 <xsd:element name="Other" type="xsd:anyType" minOccurs="0"/>  
1590 </xsd:sequence>  
1591 </xsd:complexType>
```

1592 **Schema Definition:**

```
1593 <xsd:simpleType name="LevelClassificationType">  
1594 <xsd:restriction base="xsd:string">  
1595 <xsd:enumeration value="Unclassified"/>  
1596 <xsd:enumeration value="Confidential"/>  
1597 <xsd:enumeration value="Secret"/>  
1598 <xsd:enumeration value="Top-Secret"/>  
1599 </xsd:restriction>  
1600 </xsd:simpleType>
```

1601 When it is used within a <KeyUsePolicy> element, there SHALL be only one <PermittedLevels> element within
1602 the <KeyUsePolicy> element. However, there MAY be an unbounded (unlimited) number of <PermittedLevel>
1603 elements within the <PermittedLevels> element.

1604 NOTE: It is noteworthy to mention that the absence of a <PermittedLevels> element within a <KeyUsePolicy>
1605 element specifies that applications at ALL levels are permitted the use of the symmetric key, subject to
1606 complying with all other permission clauses in the <KeyUsePolicy> element, if any.

1607 The <PermittedLevel> element, of the *LevelClassificationType*, identifies the precise MLS level at which the
1608 symmetric key in question may be used. The <PermittedLevel> SHALL contain one of the following four (4)
1609 enumerated values:

- 1610 1. Unclassified
- 1611 2. Confidential

- 1612 3. Secret
- 1613 4. Top-Secret

1614 Some examples of the <PermittedDuration> element are shown below; other parts of their enclosing elements
1615 are not shown for brevity:

1616 **Example 1 – An example of a <PermittedLevels> element specifying that the symmetric key may be used
1617 only by applications at the Confidential level:**

```
1618 <ekmi:PermittedLevels>
1619   <ekmi:PermittedLevel>Confidential</ekmi:PermittedLevel>
1620 </ekmi:PermittedLevels>
```

1621 **Example 2 – An example of a <PermittedLevels> element specifying that the symmetric key may be used
1622 only by applications at the Secret or Top-Secret level:**

```
1623 <ekmi:PermittedLevels>
1624   <ekmi:PermittedLevel>Secret</ekmi:PermittedLevel>
1625   <ekmi:PermittedLevel>Top-Secret</ekmi:PermittedLevel>
1626 </ekmi:PermittedLevels>
```

1627 2.18 Element <PermittedLocations> and <PermittedLocation>

1628 The element <PermittedLocations>, of the type *PermittedLocationsType*, is used to define the geographically
1629 physical locations where applications are permitted to use a symmetric key within a specific <Symkey> element.
1630 This element is useful only to applications and systems that have the ability to determine their Global Positioning
1631 System (GPS) location of the client device using the symmetric key.

1632 Schema Definition:

```
1633 <xsd:complexType name="PermittedLocationsType">
1634   <xsd:sequence>
1635     <xsd:element name="PermittedLocation" minOccurs="1" maxOccurs="unbounded">
1636       <xsd:complexType>
1637         <xsd:sequence>
1638           <xsd:element name="LocationName">
1639             <xsd:simpleType>
1640               <xsd:restriction base="xsd:string">
1641                 <xsd:maxLength value="256"/>
1642                 <xsd:whiteSpace value="preserve"/>
1643               </xsd:restriction>
1644             </xsd:simpleType>
1645           </xsd:element>
1646           <xsd:group
1647             ref="tns:LocationCoordinateGroup"
1648             minOccurs="0"
1649             maxOccurs="unbounded"/>
1650           <xsd:element name="Other" type="xsd:anyType" minOccurs="0"/>
1651         </xsd:sequence>
1652       </xsd:complexType>
1653     </xsd:element>
1654   </xsd:sequence>
1655 </xsd:complexType>
```

1656 Schema Definition:

```
1657 <xsd:group name="LocationCoordinateGroup">
1658   <xsd:sequence>
1659     <xsd:element name="Latitude">
1660       <xsd:simpleType>
```

```

1661         <xsd:restriction base="xsd:decimal">
1662             <xsd:totalDigits value="10"/>
1663             <xsd:fractionDigits value="7"/>
1664         </xsd:restriction>
1665     </xsd:simpleType>
1666 </xsd:element>
1667 <xsd:element name="Longitude">
1668     <xsd:simpleType>
1669         <xsd:restriction base="xsd:decimal">
1670             <xsd:totalDigits value="10"/>
1671             <xsd:fractionDigits value="7"/>
1672         </xsd:restriction>
1673     </xsd:simpleType>
1674 </xsd:element>
1675 </xsd:sequence>
1676 </xsd:group>

```

1677 When it is used within a <KeyUsePolicy> element, there SHALL be only one <PermittedLocations> element
1678 within the <KeyUsePolicy> element. However, there MAY be an unbounded (unlimited) number of
1679 <PermittedLocation> elements within the <PermittedLocations> element.

1680 NOTE: It is noteworthy to mention that the absence of a <PermittedLocations> element within a
1681 <KeyUsePolicy> element specifies that applications are permitted the use of the symmetric key at ANY physical
1682 location, subject to complying with all other permission clauses in the <KeyUsePolicy> element, if any.

1683 The <PermittedLocation> element, of the **PermittedLocationType**, identifies the precise geographical location
1684 where the symmetric key in question may be used. The <PermittedLocation> SHALL contain the following
1685 elements:

1686 1. The <LocationName> element identifies a human-readable name of the physical location. It is an XSD
1687 **String** type element, with a maximum length of 256 characters.

1688 There SHALL be only one <LocationName> element within a <PermittedLocation> element.

1690 2. An optional **LocationCoordinateGroup** which, when present, SHALL contain the following two
1691 elements:

1692 a) The <Latitude> element of XSD **Decimal** type, that identifies the horizontal coordinate location
1693 of the client device on the Earth, measured in *degrees* and expressed as a decimal with the
1694 *minutes* and *seconds* part of the measurement expressed as a single fraction.

1695 When used, there SHALL be only one <Latitude> element within the <PermittedLocation>
1696 element.

1698 b) The <Longitude> element of XSD **Decimal** type, that identifies the vertical coordinate location
1699 of the client device on the Earth, measured in *degrees* and expressed as a decimal with the
1700 *minutes* and *seconds* part of the measurement expressed as a single fraction.

1701 When used, there SHALL be only one <Longitude> element within the <PermittedLocation>
1702 element.

1704 Some examples of the <PermittedLocations> element are shown below; other parts of their enclosing elements
1705 are not shown for brevity:

1706 **Example 1 – An example of a <PermittedLocations> element specifying that the symmetric key may be
1707 used only by applications at a single named location:**

```

1708     <ekmi:PermittedLocations>
1709         <ekmi:PermittedLocation>
1710             <ekmi:LocationName>StrongAuth Server Room</ekmi:LocationName>
1711         </ekmi:PermittedLocation>
1712     </ekmi:PermittedLocations>

```

1713 **Example 2 – An example of a <PermittedLocations> element specifying that the symmetric key may be**
1714 **used only by applications at a single location at the given GPS coordinates:**

```
1715     <ekmi:PermittedLocations>  
1716         <ekmi:PermittedLocation>  
1717             <ekmi:LocationName>StrongAuth Server Room</ekmi:LocationName>  
1718             <ekmi:Latitude>37.385653 </ekmi:Latitude>  
1719             <ekmi:Longitude>-121.993192 </ekmi:Longitude>  
1720         </ekmi:PermittedLocation>  
1721     </ekmi:PermittedLocations>
```

1722 **Example 3 – An example of a <PermittedLocations> element specifying that the symmetric key may be**
1723 **used only by applications at multiple locations:**

```
1724     <ekmi:PermittedLocations>  
1725         <ekmi:PermittedLocation>  
1726             <ekmi:LocationName>Humongous Headquarters</ekmi:LocationName>  
1727         </ekmi:PermittedLocation>  
1728         <ekmi:PermittedLocation>  
1729             <ekmi:LocationName> Humongous Primary Data Center</ekmi:LocationName>  
1730             <ekmi:Latitude>37.385653 </ekmi:Latitude>  
1731             <ekmi:Longitude>-121.993192 </ekmi:Longitude>  
1732         </ekmi:PermittedLocation>  
1733         <ekmi:PermittedLocation>  
1734             <ekmi:LocationName>Humongous DR Data Center</ekmi:LocationName>  
1735             <ekmi:Latitude>68.845901 </ekmi:Latitude>  
1736             <ekmi:Longitude>11.393385 </ekmi:Longitude>  
1737         </ekmi:PermittedLocation>  
1738     </ekmi:PermittedLocations>
```

1739 **2.19 Element <PermittedNumberOfTransactions>**

1740 The element <PermittedNumberOfTransactions>, of type *PermittedNumberOfTransactionsType* is used to
1741 define the number of **encryption** transactions that applications are permitted with a symmetric key within a
1742 specific <Symkey> element, once the **SKCL** has started using the symmetric key in question. It does not limit
1743 the number of **decryption** transactions with the same symmetric key.

1744 **Schema Definition:**

```
1745     <xsd:simpleType name="PermittedNumberOfTransactionsType">  
1746         <xsd:restriction base="xsd:positiveInteger">  
1747             <xsd:maxInclusive value="18446744073709551615"/>  
1748         </xsd:restriction>  
1749     </xsd:simpleType>
```

1750 When it is used within a <KeyUsePolicy> element, there SHALL be only one <PermittedNumberOfTransactions>
1751 element within the <KeyUsePolicy> element.

1752 NOTE: It is noteworthy to mention that the absence of a <PermittedNumberOfTransactions> element within a
1753 <KeyUsePolicy> element specifies that applications are permitted the use of the symmetric key for an unlimited
1754 number of encryption transactions, subject to complying with all other permission clauses in the <KeyUsePolicy>
1755 element, if any.

1756 The <PermittedNumberOfTransactions> element, of the XSD *positiveInteger* type, identifies the precise number
1757 of encryption transactions that may be performed by the symmetric key in question, once the key has been used
1758 by conforming **SKCL** implementations for the first time.

1759 Some examples of the <PermittedNumberOfTransactions> element are shown below; other parts of their
1760 enclosing elements are not shown for brevity:

1761 **Example 1 – An example of a <PermittedNumberOfTransactions> element specifying that the symmetric**
1762 **key may be used only for a single encryption transaction by an SKCL:**

```
1763 <ekmi:PermittedNumberOfTransactions>1</ekmi:PermittedNumberOfTransactions>
```

1764 **Example 2 – An example of a <PermittedNumberOfTransactions> element specifying that the symmetric**
1765 **key may be used only for 100 transactions by an SKCL:**

```
1766 <ekmi:PermittedNumberOfTransactions>100</ekmi:PermittedNumberOfTransactions>
```

1767 **2.20 Element <PermittedTimes> and <PermittedTime>**

1768 The element <PermittedTimes>, of the type *PermittedTimesType* and its only child-element <PermittedTime>,
1769 which is an anonymous XSD *ComplexType*, are used to define sets of times during the day between which
1770 applications are permitted to use a symmetric key within a specific <Symkey> element.

1771 **Schema Definition:**

```
1772 <xsd:complexType name="PermittedTimesType">  
1773 <xsd:sequence>  
1774 <xsd:element name="PermittedTime" minOccurs="0" maxOccurs="unbounded">  
1775 <xsd:complexType>  
1776 <xsd:sequence>  
1777 <xsd:element name="StartTime">  
1778 <xsd:simpleType>  
1779 <xsd:restriction base="xsd:time">  
1780 <xsd:pattern value="\p{Nd}{2}:\p{Nd}{2}:\p{Nd}{2}"/>  
1781 </xsd:restriction>  
1782 </xsd:simpleType>  
1783 </xsd:element>  
1784 <xsd:element name="EndTime">  
1785 <xsd:simpleType>  
1786 <xsd:restriction base="xsd:time">  
1787 <xsd:pattern value="\p{Nd}{2}:\p{Nd}{2}:\p{Nd}{2}"/>  
1788 </xsd:restriction>  
1789 </xsd:simpleType>  
1790 </xsd:element>  
1791 </xsd:sequence>  
1792 </xsd:complexType>  
1793 </xsd:element>  
1794 </xsd:sequence>  
1795 </xsd:complexType>
```

1796 When it is used within a <KeyUsePolicy> element, there SHALL be only one <PermittedTimes> element within
1797 the <KeyUsePolicy> element. However, there MAY be an unbounded (unlimited) number of <PermittedTime>
1798 elements within a <PermittedTimes> element.

1799 **NOTE:** It is noteworthy to mention that the absence of a <PermittedTimes> element within a <KeyUsePolicy>
1800 element specifies that applications are permitted the use of the symmetric key at any time of the day, subject to
1801 complying with all other permission clauses in the <KeyUsePolicy> element, if any.

1802 The <PermittedTime> element identifies an individual set of times between which application are permitted to use
1803 the symmetric key in question. The <PermittedTime> element consists of the following sub-elements:

- 1804 1. The <StartTime> element identifies the date from which applications may start using the symmetric key
1805 in question. It is an XSD *Time* type that MUST be specified in a specific pattern (see examples) where
1806 the first two digits specify the hour, the second two digits specify the minutes and the last two digits
1807 specify the seconds in a 24 hour format.

1808 There SHALL be only one <StartTime> element within a <PermittedTime> element.
1809
1810

1811 Conforming **SKCL** implementations SHALL NOT start using the symmetric before the onset of the
1812 <StartTime> on the client machine.

1813 2. The <EndTime> element identifies the time until which applications may use the symmetric key in
1814 question. It is an XSD *Time* type that MUST be specified in a specific pattern (see examples) where the
1815 first two digits specify the hour, the second two digits specify the minutes and the last two digits specify
1816 the seconds in a 24 hour format.

1817
1818 There SHALL be only one <EndTime> element within a <PermittedTime> element.

1819
1820 Conforming **SKCL** implementations SHALL NOT use the symmetric after the end of the <EndTime> on
1821 the client machine.

1822 Some examples of the <PermittedTimes> element are shown below; other parts of their enclosing elements are
1823 not shown for brevity:

1824 **Example 1 – An example of a <PermittedTimes> element with a single<PermittedTime> element. The**
1825 **<StartTime> specifies 9:00AM on the client machine while the <EndTime> specifies 5:00PM:**

```
1826 <ekmi:PermittedTimes>  
1827 <ekmi:PermittedTime>  
1828 <ekmi:StartTime>09:00:00</ekmi:StartTime>  
1829 <ekmi:EndTime>17:00:00</ekmi:EndTime>  
1830 </ekmi:PermittedTime>  
1831 </ekmi:PermittedTimes>
```

1832 **Example 2 – An example of a <PermittedTimes> element with two <PermittedTime> elements. For the**
1833 **first <PermittedTime> element , the <StartTime> element specifies 6:00AM while the <EndTime> element**
1834 **specifies 12:00 Noon. For the second <PermittedTime> element, the <StartTime> element specifies 3:00**
1835 **PM in the afternoon, while the <EndTime> element specifies 7:00PM in the evening. This policy might**
1836 **imply that a symmetric key with this <PermittedTimes> element cannot be used during a lunch break of**
1837 **12:00 Noon to 3:00PM:**

```
1838 <ekmi:PermittedTimes>  
1839 <ekmi:PermittedTime>  
1840 <ekmi:StartTime>06:00:00</ekmi:StartTime>  
1841 <ekmi:EndTime>12:00:00</ekmi:EndTime>  
1842 </ekmi:PermittedTime>  
1843 <ekmi:PermittedTime>  
1844 <ekmi:StartTime>15:00:00</ekmi:StartTime>  
1845 <ekmi:EndTime>19:00:00</ekmi:EndTime>  
1846 </ekmi:PermittedTime>  
1847 </ekmi:PermittedTimes>
```

1848 2.21 Element <PermittedUses> and <PermittedUse>

1849 The element <PermittedUses>, of the type *PermittedUsesType*, is used to define the specific ways in which
1850 applications are permitted to use a symmetric key within a specific <Symkey> element.

1851 **Schema Definition:**

```
1852 <xsd:complexType name="PermittedUsesType" mixed="true">  
1853 <xsd:sequence>  
1854 <xsd:element name="PermittedUse" minOccurs="0" maxOccurs="unbounded">  
1855 <xsd:simpleType>  
1856 <xsd:restriction base="xsd:string">  
1857 <xsd:maxLength value="256"/>  
1858 <xsd:whiteSpace value="preserve"/>  
1859 </xsd:restriction>  
1860 </xsd:simpleType>  
1861 </xsd:element>
```

```
1862     <xsd:element name="Other" type="xsd:anyType" minOccurs="0"/>
1863     </xsd:sequence>
1864 </xsd:complexType>
```

1865 When it is used within a <KeyUsePolicy> element, there SHALL be only one <PermittedUses> element within the
1866 <KeyUsePolicy> element. However, there MAY be an unbounded (unlimited) number of <PermittedUse>
1867 elements within the <PermittedUses> element.

1868 NOTE: It is noteworthy to mention that the absence of a <PermittedUses> element within a <KeyUsePolicy>
1869 element specifies that applications are permitted the use of the symmetric key for any purpose, subject to
1870 complying with all other permission clauses in the <KeyUsePolicy> element, if any.

1871 Some examples of the <PermittedUses> element are shown below; other parts of their enclosing elements are
1872 not shown for brevity:

1873 **Example 1 – An example of a <PermittedUses> element specifying that the symmetric key may be used**
1874 **only by VPN applications for session encryption keys:**

```
1875     <ekmi:PermittedUses>
1876         <ekmi:PermittedUse>VPN</ekmi:PermittedUse>
1877     </ekmi:PermittedUses>
```

1878 **Example 2 – An example of a <PermittedUses> element specifying that the symmetric key may be used**
1879 **only by applications on laptops and Personal Digital Assistants (PDA):**

```
1880     <ekmi:PermittedUses>
1881         <ekmi:PermittedUse>Laptop</ekmi:PermittedUse>
1882         <ekmi:PermittedUse>PDA</ekmi:PermittedUse>
1883     </ekmi:PermittedUses>
```

1884 2.22 Element <KeyCachePolicyRequest>

1885 The <KeyCachePolicyRequest> element is used to request a key-cache policy from the SKS server , so the
1886 client may know if and how to cache symmetric keys locally.

1887 While it is a top-level element within this specification, a <SymkeyRequest> element MUST be enclosed within a
1888 **SOAP Body** element of a **SOAP Envelope** to conform to the security requirements of this specification. The
1889 **SOAP Header** of the **SOAP Envelope** MUST enclose a **Security** element conforming to [WSS] with a
1890 **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
1891 in [WSS] to form a well-formed, secure message.
1892

1893 Schema Definition:

```
1894     <xsd:element name="KeyCachePolicyRequest">
1895         <xsd:complexType>
1896             <xsd:annotation>
1897                 <xsd:documentation>
1898                     No elements/attributes are defined for KeyCachePolicyRequest.
1899                 </xsd:documentation>
1900             </xsd:annotation>
1901         </xsd:complexType>
1902     </xsd:element>
```

1903 The <KeyCachePolicyRequest> has no child elements. The SOAP Header of the signed request provides the
1904 SKS server with all the information it needs to process the request: the identity of the requester, strong
1905 authentication and message integrity of the request.

1906 Some examples of the use of the <SymkeyRequest> element are as follows:

1907 **Example 1 – An example of a <KeyCachePolicyRequest>; the surrounding SOAP envelope is not**
1908 **displayed here for brevity:**


```
1909     <ekmi:KeyCachePolicyRequest
1910         xmlns:ekmi="http://docs.oasis-open.org/ekmi/2008/01"/>
```

1911 2.23 Element <KeyCachePolicyResponse>

1912 The <KeyCachePolicyResponse> element is the response sent by an **SKS** Server to a client that requested a
1913 key-cache policy through a <KeyCachePolicyRequest>. The <KeyCachePolicyResponse> contains
1914 policy elements, which define rules that conforming implementations of the **SKCL** MUST adhere to when
1915 caching symmetric keys sent by the **SKS** Server.

1916 Schema Definition:

```
1917     <xsd:element name="KeyCachePolicyResponse">
1918         <xsd:complexType>
1919             <xsd:sequence>
1920                 <xsd:element
1921                     name="KeyCachePolicy"
1922                     type="ekmi:KeyCachePolicyType"
1923                     minOccurs="1" maxOccurs="unbounded"/>
1924             </xsd:sequence>
1925         </xsd:complexType>
1926     </xsd:element>
```

1927 The <KeyCachePolicyResponse> element consists of a minimum of one, but an unbounded (unlimited)
1928 number of <KeyCachePolicy> children elements.

1929 2.24 Element <KeyCachePolicy>

1930 The <KeyCachePolicy> element contains policy elements, which define rules that conforming implementations
1931 of the **SKCL** MUST adhere to when caching symmetric keys sent by the **SKS** Server.

1932 Schema Definition:

```
1933     <xsd:element name="KeyCachePolicyResponse">
1934         <xsd:complexType>
1935             <xsd:sequence>
1936                 <xsd:element
1937                     name="KeyCachePolicy"
1938                     type="ekmi:KeyCachePolicyType"
1939                     minOccurs="1" maxOccurs="unbounded"/>
1940             </xsd:sequence>
1941         </xsd:complexType>
1942     </xsd:element>

1943     <xsd:complexType name="KeyCachePolicyType" mixed="true">
1944         <xsd:sequence>
1945             <xsd:element name="KeyCachePolicyID" type="tns:TwoPartIDType"/>
1946             <xsd:element name="PolicyName">
1947                 <xsd:simpleType>
1948                     <xsd:restriction base="xsd:string">
1949                         <xsd:maxLength value="255"/>
1950                         <xsd:whiteSpace value="preserve"/>
1951                     </xsd:restriction>
1952                 </xsd:simpleType>
1953             </xsd:element>
1954             <xsd:element name="Description" nillable="true">
1955                 <xsd:simpleType>
1956                     <xsd:restriction base="xsd:string">
1957                         <xsd:maxLength value="2048"/>
```

```

1958         <xsd:whiteSpace value="preserve"/>
1959         </xsd:restriction>
1960     </xsd:simpleType>
1961 </xsd:element>
1962 <xsd:element name="KeyClass" type="tns:KeyClassType"/>
1963 <xsd:element name="StartDate" type="xsd:dateTime"/>
1964 <xsd:element name="EndDate" type="xsd:dateTime" nillable="true"/>
1965 <xsd:element name="PolicyCheckInterval">
1966     <xsd:simpleType>
1967         <xsd:restriction base="xsd:nonNegativeInteger">
1968             <xsd:minInclusive value="0"/>
1969             <xsd:maxInclusive value="2592000"/>
1970         </xsd:restriction>
1971     </xsd:simpleType>
1972 </xsd:element>
1973 <xsd:element name="Status" type="tns:StatusType"/>
1974 <xsd:element
1975     name="NewKeysCacheDetail"
1976     type="tns:KeyCacheDetailType"
1977     minOccurs="0"/>
1978 <xsd:element
1979     name="UsedKeysCacheDetail"
1980     type="tns:KeyCacheDetailType"
1981     minOccurs="0"/>
1982 </xsd:sequence>
1983 </xsd:complexType>

```

1984 The <KeyCachePolicy> element is of the **KeyCachePolicyType** and consists of the following child elements:

1985 1. <KeyCachePolicyID> [Required]

1986
1987 The <KeyCachePolicyID> element, of type **TwoPartIDType**, identifies the unique policy object within
1988 the **SKMS**. There SHALL be only one <KeyCachePolicyID> element within a <KeyCachePolicy>
1989 element.

1990 The **TwoPartIDType** is specified in Section 2.8.

1992 2. <PolicyName> [Required]

1993
1994 The <PolicyName> element, of type XSD **String**, with a maximum length of 255 characters, identifies
1995 a unique name given to this <KeyCachePolicy>. There SHALL be only one <PolicyName> element
1996 within a <KeyCachePolicy> element.

1997 3. <Description> [Required]

1998
1999 The KeyCachePolicy element, of type XSD **String**, with a maximum length of 2048 characters,
2000 provides a human-readable description of this policy. There SHALL be only one <Description>
2001 element within a <KeyCachePolicy> element.

2002 The <Description> MAY be an empty element, but MUST exist within the <KeyCachePolicy>
2003 element.

2005 4. <KeyClass> [Required]

2006
2007 This element of type **KeyClassType** identifies the key-class of the symmetric key to which this policy
2008 applies.

2009 5. <StartDate> [Required]

2010
2011 The <StartDate> element, of type XSD **dateTime**, specifies the date and time at which this policy

2012 becomes effective. There SHALL be only one <StartDate> element within a <KeyCachePolicy>
2013 element.

2014 6. <EndDate> [Required]
2015
2016 The <EndDate> element , of type XSD *dateTime*, specifies the date and time at which this policy
2017 expires. There SHALL be only one <EndDate> element within a <KeyCachePolicy> element.
2018
2019 The <EndDate> MAY be an empty element, but MUST exist within the <KeyCachePolicy> element.

2020 7. <PolicyCheckInterval> [Required]
2021
2022 The <PolicyCheckInterval> element , of type XSD *nonNegativeInteger*, specifies the frequency at
2023 which the client SHALL check the **SKS** server for updates to this policy. This frequency is specified in
2024 seconds and SHALL NOT exceed 2592000 seconds (30 calendar days). There SHALL be only one
2025 <PolicyCheckInterval> element within a <KeyCachePolicy> element.

2026 8. <Status> [Required]
2027
2028 The <Status> element, of type *StatusType*, identifies the current status of this policy within the SKMS.
2029 There SHALL be only one <Status> element within a <KeyCachePolicy> element.
2030
2031 The *StatusType* is specified in Section 2.11.

2032 9. <NewKeysCacheDetail> [Required]
2033
2034 The <NewKeysCacheDetail> element, of type *KeyCacheDetailType*, defines how many new (as yet
2035 unused for any encryption transaction) symmetric keys a client may cache, and for how long. It is the
2036 responsibility of the conforming **SKCL** implementation to enforce these rules.
2037
2038 The absence of the <NewKeysCacheDetail> element implies that new symmetric keys SHALL NEVER
2039 be cached on the client. New keys may be cached only when this element exists, and SHALL conform
2040 to the rules specified in this element.
2041
2042 When it exists, there SHALL be only one <NewKeysCacheDetail> element in a <KeyCachePolicy>
2043 element.
2044
2045 The *KeyCacheDetailType* is specified in Section 2.22.

2046 10. <UsedKeysCacheDetail> [Required]
2047
2048 The <UsedKeysCacheDetail> element, of type *KeyCacheDetailType*, defines how many used
2049 symmetric keys a client may cache, and for how long. It is the responsibility of the conforming **SKCL**
2050 implementation to enforce these rules.
2051
2052 The absence of the <UsedKeysCacheDetail> element implies that used symmetric keys SHALL
2053 NEVER be cached on the client. Used keys may be cached only when this element exists, and SHALL
2054 conform to the rules specified in this element.
2055
2056 When it exists, there SHALL be only one <UsedKeysCacheDetail> element in a <KeyCachePolicy>
2057 element.
2058
2059 The *KeyCacheDetailType* is specified in Section 2.22.

2060 Some examples of the <KeyUsePolicy> element are as follows.

2061 **Example 1 – A <KeyCachePolicy> that is valid between January 01, 2008 and December 31, 2008. It**
2062 **requires the client to check for policy updates every day and allows 3 new and 3 used keys to be cached**
2063 **for up to 90 days:**

```

2064 <ekmi:KeyCachePolicy>
2065   <ekmi:KeyCachePolicyID>10514-17</ekmi:KeyCachePolicyID>
2066   <ekmi:PolicyName>
2067     Corporate Laptop Symmetric Key Caching Policy
2068   </ekmi:PolicyName>
2069   <ekmi:Description>
2070     This policy defines how company-issued laptops will manage
2071     symmetric keys used for file/disk encryption in each laptop's
2072     local cache. This policy must be used by all laptops that
2073     use the company EKMI.
2074   </ekmi:Description>
2075   <ekmi:StartDate>2008-01-01T00:00:01.0</ekmi:StartDate>
2076   <ekmi:EndDate>2008-12-31T24:00:00.0</ekmi:EndDate>
2077   <ekmi:PolicyCheckInterval>86400</ekmi:PolicyCheckInterval>
2078   <ekmi:Status>Active</ekmi:Status>
2079   <ekmi:NewKeysCacheDetail>
2080     <ekmi:MaximumKeys>3</ekmi:MaximumKeys>
2081     <ekmi:MaximumDuration>7776000</ekmi:MaximumDuration>
2082   </ekmi:NewKeysCacheDetail>
2083   <ekmi:UsedKeysCacheDetail>
2084     <ekmi:MaximumKeys>3</ekmi:MaximumKeys>
2085     <ekmi:MaximumDuration>7776000</ekmi:MaximumDuration>
2086   </ekmi:UsedKeysCacheDetail>
2087 </ekmi:KeyCachePolicy>

```

2088 **Example 2 – A <KeyCachePolicy> that is effective starting January 01, 2008 and never expires. It does**
2089 **NOT permit any caching of symmetric keys through the absence of the detail elements on caching:**

```

2090 <ekmi:KeyCachePolicy>
2091   <ekmi:KeyCachePolicyID>10514-1</ekmi:KeyCachePolicyID>
2092   <ekmi:PolicyName>
2093     No Caching Policy
2094   </ekmi:PolicyName>
2095   <ekmi:Description>
2096     This policy is for high-risk, always-connected machines on the
2097     network, which will never cache symmetric keys locally. This
2098     policy never expires (but checks monthly for any updates).
2099   </ekmi:Description>
2100   <ekmi:StartDate>2008-01-01T00:00:01.0</ekmi:StartDate>
2101   <ekmi:EndDate>1969-01-01T00:00:00.0</ekmi:EndDate>
2102   <ekmi:PolicyCheckInterval>2592000</ekmi:PolicyCheckInterval>
2103   <ekmi:Status>Active</ekmi:Status>
2104 </ekmi:KeyCachePolicy>

```

2105 **2.25 Type *KeyCacheDetailType***

2106 The ***KeyCacheDetailType*** type allows SKS servers to specify precisely how many symmetric keys MAY be
2107 cached on the client machine, and for how long.

2108 **Schema Definition:**

```

2109 <xsd:complexType name="KeyCacheDetailType">
2110   <xsd:sequence>
2111     <xsd:element name="MaximumKeys" minOccurs="1">
2112       <xsd:simpleType>
2113         <xsd:restriction base="xsd:integer">
2114           <xsd:minInclusive value="0"/>
2115           <xsd:maxInclusive value="18446744073709551615"/>
2116         </xsd:restriction>
2117       </xsd:simpleType>

```

```

2118         </xsd:element>
2119         <xsd:element name="MaximumDuration" minOccurs="1">
2120             <xsd:simpleType>
2121                 <xsd:restriction base="xsd:integer">
2122                     <xsd:minInclusive value="0"/>
2123                     <xsd:maxInclusive value="18446744073709551615"/>
2124                 </xsd:restriction>
2125             </xsd:simpleType>
2126         </xsd:element>
2127     </xsd:sequence>
2128 </xsd:complexType>

```

2129 The **KeyCacheDetailType** consists of the following child elements:

2130 1. <MaximumKeys> [Required]

2131
2132 The <MaximumKeys> element, of type XSD **Integer**, specifies the maximum number of symmetric keys
2133 that MAY be cached on a client machine. It SHALL be a positive number between the values 0 and
2134 18446744073709551615. There SHALL be only one <MaximumKeys> element within an element that
2135 uses the **KeyCacheDetailType**.

2136 2. <MaximumDuration> [Required]

2137
2138 The <MaximumDuration> element, of type XSD **Integer**, specifies the maximum number of seconds
2139 that symmetric keys MAY be cached on a client machine. It SHALL be a positive number between the
2140 values 0 and 18446744073709551615. There SHALL be only one <MaximumDuration> element
2141 within an element that uses the **KeyCacheDetailType**.

2142 Examples of the **KeyCacheDetailType** when used in the <KeyCachePolicy> element are as follows.

2143 **Example 1 – A <KeyCachePolicy> that is valid between January 01, 2008 and December 31, 2008. It**
2144 **requires the client to check for policy updates every day and allows 3 new and 3 used keys to be cached**
2145 **for up to 90 days:**

```

2146 <ekmi:KeyCachePolicy>
2147     <ekmi:KeyCachePolicyID>10514-17</ekmi:KeyCachePolicyID>
2148     <ekmi:PolicyName>
2149         Corporate Laptop Symmetric Key Caching Policy
2150     </ekmi:PolicyName>
2151     <ekmi:Description>
2152         This policy defines how company-issued laptops will manage
2153         symmetric keys used for file/disk encryption in their local
2154         cache. This policy must be used by all laptops that use
2155         the company EKMI.
2156     </ekmi:Description>
2157     <ekmi:StartDate>2008-01-01T00:00:01.0</ekmi:StartDate>
2158     <ekmi:EndDate>2008-12-31T24:00:00.0</ekmi:EndDate>
2159     <ekmi:PolicyCheckInterval>86400</ekmi:PolicyCheckInterval>
2160     <ekmi:Status>Active</ekmi:Status>
2161     <ekmi:NewKeysCacheDetail>
2162         <ekmi:MaximumKeys>3</ekmi:MaximumKeys>
2163         <ekmi:MaximumDuration>7776000</ekmi:MaximumDuration>
2164     </ekmi:NewKeysCacheDetail>
2165     <ekmi:UsedKeysCacheDetail>
2166         <ekmi:MaximumKeys>3</ekmi:MaximumKeys>
2167         <ekmi:MaximumDuration>7776000</ekmi:MaximumDuration>
2168     </ekmi:UsedKeysCacheDetail>
2169 </ekmi:KeyCachePolicy>

```

2170 **Example 1 – A <KeyCachePolicy> that is valid between January 01, 2008 and December 31, 2008. It**
2171 **requires the client to check for policy updates every day and allows 1 new and 0 used keys to be cached**
2172 **for upto 15 days:**

```
2173 <ekmi:KeyCachePolicy>
2174   <ekmi:KeyCachePolicyID>10514-17</ekmi:KeyCachePolicyID>
2175   <ekmi:PolicyName>
2176     Corporate Laptop Symmetric Key Caching Policy
2177   </ekmi:PolicyName>
2178   <ekmi:Description>
2179     This policy defines how company-issued laptops will manage
2180     symmetric keys used for file/disk encryption in each laptop's
2181     local cache. This policy must be used by all laptops that
2182     use the company EKMI.
2183   </ekmi:Description>
2184   <ekmi:StartDate>2008-01-01T00:00:01.0</ekmi:StartDate>
2185   <ekmi:EndDate>2008-12-31T24:00:00.0</ekmi:EndDate>
2186   <ekmi:PolicyCheckInterval>86400</ekmi:PolicyCheckInterval>
2187   <ekmi:Status>Active</ekmi:Status>
2188   <ekmi:NewKeysCacheDetail>
2189     <ekmi:MaximumKeys>1</ekmi:MaximumKeys>
2190     <ekmi:MaximumDuration>1296000</ekmi:MaximumDuration>
2191   </ekmi:NewKeysCacheDetail>
2192   <ekmi:UsedKeysCacheDetail>
2193     <ekmi:MaximumKeys>0</ekmi:MaximumKeys>
2194     <ekmi:MaximumDuration>1296000</ekmi:MaximumDuration>
2195   </ekmi:UsedKeysCacheDetail>
2196 </ekmi:KeyCachePolicy>
```

2197 **Appendix A. Acknowledgments**

2198 The following individuals have participated in the creation of this specification and are gratefully
2199 acknowledged

2200 **Participants:**

2201 •

2202

2203

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.

2204

2205

2206

2207

Appendix C. Non-Normative Text

2208

Appendix D. SKSML Error Codes and Error Messages