



1
2
3
4

Document Number: DSP1096

Date: 2011-09-16

Version: 1.0.0

5 **Certificate Management Profile**

6 **Document Type: Specification**
7 **Document Status: DMTF Standard**
8 **Document Language: en-US**
9

10 Copyright notice

11 Copyright © 2011 Distributed Management Task Force, Inc. (DMTF). All rights reserved.

12 DMTF is a not-for-profit association of industry members dedicated to promoting enterprise and systems
13 management and interoperability. Members and non-members may reproduce DMTF specifications and
14 documents, provided that correct attribution is given. As DMTF specifications may be revised from time to
15 time, the particular version and release date should always be noted.

16 Implementation of certain elements of this standard or proposed standard may be subject to third party
17 patent rights, including provisional patent rights (herein "patent rights"). DMTF makes no representations
18 to users of the standard as to the existence of such rights, and is not responsible to recognize, disclose,
19 or identify any or all such third party patent right, owners or claimants, nor for any incomplete or
20 inaccurate identification or disclosure of such rights, owners or claimants. DMTF shall have no liability to
21 any party, in any manner or circumstance, under any legal theory whatsoever, for failure to recognize,
22 disclose, or identify any such third party patent rights, or for such party's reliance on the standard or
23 incorporation thereof in its product, protocols or testing procedures. DMTF shall have no liability to any
24 party implementing such standard, whether such implementation is foreseeable or not, nor to any patent
25 owner or claimant, and shall have no liability or responsibility for costs or losses incurred if a standard is
26 withdrawn or modified after publication, and shall be indemnified and held harmless by any party
27 implementing the standard from any and all claims of infringement by a patent owner for such
28 implementations.

29 For information about patents held by third-parties which have notified the DMTF that, in their opinion,
30 such patent may relate to or impact implementations of DMTF standards, visit
31 <http://www.dmtf.org/about/policies/disclosures.php>.

32

Contents

34	Foreword	6
35	Introduction	7
36	1 Scope	9
37	2 Normative references	9
38	3 Terms and definitions	9
39	4 Symbols and abbreviated terms	10
40	5 Synopsis	11
41	6 Description	11
42	6.1 Key store	12
43	6.2 Public key infrastructure	13
44	6.3 Authorization	13
45	7 Implementation	14
46	7.1 Key store	14
47	7.2 Certificate management service	14
48	7.3 Certificate chain	15
49	7.4 Authorization	15
50	8 Methods	16
51	8.1 CIM_CertificateManagementService.ImportPublicPrivateKeyPair()	16
52	8.2 CIM_CertificateManagementService.CreateKeystore()	18
53	8.3 CIM_CertificateManagementService.CreateCertificateSigningRequest()	19
54	8.4 CIM_CertificateManagementService.CreateSelfSignedCertificate()	21
55	8.5 CIM_CertificateManagementService.ImportEncodedCertificates()	23
56	8.6 CIM_CertificateManagementService.ImportCertificates()	24
57	8.7 CIM_CertificateManagementService.ExportEncodedCertificates()	25
58	8.8 CIM_CertificateManagementService.ApplyCRL()	27
59	8.9 CIM_CertificateManagementService.ApplyDecodedCRL()	28
60	8.10 Profile conventions for operations	29
61	8.11 CIM_CertificateManagementCapabilities	29
62	8.12 CIM_CertificateManagementService	29
63	8.13 CIM_CredentialContext	30
64	8.14 CIM_ConcreteDependency (CIM_Keystore)	30
65	8.15 CIM_ConcreteDependency (CIM_X509Certificate)	30
66	8.16 CIM_ElementCapabilities	31
67	8.17 CIM_HostedService	31
68	8.18 CIM_Keystore	31
69	8.19 CIM_MemberOfCollection	31
70	8.20 CIM_OwningCollectionElement	32
71	8.21 CIM_RegisteredProfile	32
72	8.22 CIM_ServiceAffectsElement	32
73	8.23 CIM_UnsignedCredential	32
74	8.24 CIM_X509Certificate	32
75	8.25 CIM_X509CRL	33
76	8.26 CIM_Identity	33
77	8.27 CIM_AssociatedPrivilege	33
78	8.28 CIM_ConcreteDependency	34
79	9 Use cases	34
80	9.1 Profile registration	34
81	9.2 Simple certificate management	35
82	9.3 Keystore	36
83	9.4 Import asymmetric key	37
84	9.5 CSR and self-signed certificate	38

85	9.6	Import and export of certificates	39
86	9.7	CRL	40
87	10	CIM elements	42
88	10.1	CIM_CertificateManagementCapabilities	43
89	10.2	CIM_CertificateManagementService	44
90	10.3	CIM_CredentialContext.....	44
91	10.4	CIM_ConcreteDependency (CIM_Keystore)	45
92	10.5	CIM_ConcreteDependency (CIM_X509Certificate).....	45
93	10.6	CIM_ConcreteDependency	45
94	10.7	CIM_ElementCapabilities	46
95	10.8	CIM_HostedService.....	46
96	10.9	CIM_Keystore	46
97	10.10	CIM_ServiceAffectsElement (CIM_Credential)	47
98	10.11	CIM_MemberOfCollection	47
99	10.12	CIM_OwningCollectionElement.....	48
100	10.13	CIM_RegisteredProfile.....	48
101	10.14	CIM_ServiceAffectsElement (CIM_Keystore).....	48
102	10.15	CIM_UnsignedCredential.....	49
103	10.16	CIM_X509Certificate.....	49
104	10.17	CIM_X509CRL.....	50
105	10.18	CIM_AssociatedPrivilege.....	50
106	ANNEX A (informative)	Change Log.....	51
107			
108	Figures		
109	Figure 1 – Certificate Management Profile: Class diagram		12
110	Figure 2 – Profile registration.....		35
111	Figure 3 – Simple certificate management		36
112	Figure 4 – Key store.....		37
113	Figure 5 – Importing public/private key pair.....		38
114	Figure 6 – Creating CSR and self-signed certificate.....		39
115	Figure 7 – Import and export of certificates		40
116	Figure 8 – Before CRL application.....		41
117	Figure 9 – After CRL application.....		42
118			
119	Tables		
120	Table 1 – Referenced profiles.....		11
121	Table 2 – CIM_AssociatedPrivilege.Activities mapping to PKI Credential Instance operations.....		15
122	Table 3 – CIM_AssociatedPrivilege.Activities mapping to PKI Credential Instance operations.....		16
123	Table 4 – CIM_CertificateManagementService.ImportPublicPrivateKeyPair() method: Return code values		16
124	Table 5 – CIM_CertificateManagementService.ImportPublicPrivateKeyPair() method: Parameters.....		17
125	Table 6 – CIM_CertificateManagementService.CreateKeystore() method: Return code values		18
126	Table 7 – CIM_CertificateManagementService.CreateKeystore() method: Parameters.....		18
127	Table 8 – CIM_CertificateManagementService.CreateCertificateSigningRequest() method: Return code		
128	values.....		19
129	Table 9 – CIM_CertificateManagementService.CreateCertificateSigningRequest() method: Parameters		19
130	Table 10 – CIM_CertificateManagementService.CreateSelfSignedCertificate() method: Return code		
131	values.....		21
132	Table 11 – CIM_CertificateManagementService.CreateSelfSignedCertificate() method: Parameters		21
133	Table 12 – CIM_CertificateManagementService.ImportEncodedCertificates() method: Return code		
134	values.....		23

135 Table 13 – CIM_CertificateManagementService.ImportEncodedCertificates() method: Parameters 23

136 Table 14 – CIM_CertificateManagementService.ImportCertificates() method: Return code values 25

137 Table 15 – CIM_CertificateManagementService.ImportCertificates() method: Parameters 25

138 Table 16 – CIM_CertificateManagementService.ExportEncodedCertificates() method: Return code

139 values 26

140 Table 17 – CIM_CertificateManagementService.ExportEncodedCertificates() method: Parameters 26

141 Table 18 – CIM_CertificateManagementService.ApplyCRL() method: Return code values 27

142 Table 19 – CIM_CertificateManagementService.ApplyCRL() method: Parameters 27

143 Table 20 – CIM_CertificateManagementService.ApplyDecodedCRL() method: Return code values 28

144 Table 21 – CIM_CertificateManagementService.ApplyDecodedCRL() method: Parameters 28

145 Table 22 – Operations: CIM_CredentialContext 30

146 Table 23 – Operations: CIM_ConcreteDependency 30

147 Table 24 – Operations: CIM_ConcreteDependency 30

148 Table 25 – Operations: CIM_ElementCapabilities 31

149 Table 26 – Operations: CIM_HostedService 31

150 Table 27 – Operations: CIM_MemberOfCollection 31

151 Table 28 – Operations: CIM_OwningCollectionElement 32

152 Table 29 – Operations: CIM_ServiceAffectsElement 32

153 Table 30 – Operations: CIM_X509Certificate 33

154 Table 31 – Operations: CIM_AssociatedPrivilege 34

155 Table 32 – Operations: CIM_ConcreteDependency 34

156 Table 33 – CIM elements: Certificate Management Profile 42

157 Table 34 – Class: CIM_CertificateManagementCapabilities 43

158 Table 35 – Class: CIM_CertificateManagementService 44

159 Table 36 – Class: CIM_CredentialContext 45

160 Table 37 – Class: CIM_ConcreteDependency 45

161 Table 38 – Class: CIM_ConcreteDependency 45

162 Table 39 – CIM_ConcreteDependency 46

163 Table 40 – CIM_ElementCapabilities 46

164 Table 41 – Class: CIM_HostedService 46

165 Table 42 – Class: CIM_Keystore 46

166 Table 43 – Class: CIM_ServiceAffectsElement (CIM_Credential) 47

167 Table 44 – Class: CIM_MemberOfCollection 47

168 Table 45 – Class: CIM_OwningCollectionElement 48

169 Table 46 – Class: CIM_RegisteredProfile 48

170 Table 47 – Class: CIM_ServiceAffectsElement (CIM_Keystore) 49

171 Table 48 – Class: CIM_UnsignedCredential 49

172 Table 49 – Class: CIM_X509Certificate 49

173 Table 50 – Class: CIM_X509CRL 50

174 Table 51 – Class: CIM_AssociatedPrivilege 50

175

176

Foreword

177 The *Certificate Management Profile* (DSP1096) was prepared by the Security Working Group of DMTF.

178 DMTF is a not-for-profit association of industry members dedicated to promoting enterprise and systems
179 management and interoperability. For information about the DMTF, see <http://www.dmtf.org>.

180 **Acknowledgments**

181 The DMTF acknowledges the following individuals for their contributions to this document:

- 182 • Khachatur Papanyan – Dell
- 183 • George Ericson – EMC
- 184 • Jeff Hilland – HP
- 185 • David Hines – Intel
- 186 • Hemal Shah – Broadcom
- 187 • Sharon L. Smith – Intel

188

189

190

Introduction

191 The information in this specification is intended to be sufficient for a provider or consumer of this data to
192 identify unambiguously the classes, properties, methods, and values that are mandatory to be
193 instantiated and manipulated to represent and manage users and groups that are modeled using the
194 DMTF Common Information Model (CIM) core and extended model definitions.

195 The target audience for this specification is implementers who are writing CIM-based providers or
196 consumers of management interfaces that represent the component described in this document.

197 Document conventions

198 Typographical conventions

199 The following typographical conventions are used in this document:

- 200 • Document titles are marked in *italics*.
- 201 • Important terms that are used for the first time are marked in *italics*.
- 202 • ABNF rules are in `monospaced font`.

203 ABNF usage conventions

204 Format definitions in this document are specified using ABNF (see [RFC5234](#)), with the following
205 deviations:

- 206 • Literal strings are to be interpreted as case-sensitive Unicode characters, as opposed to the
207 definition in [RFC5234](#) that interprets literal strings as case-insensitive US-ASCII characters.

208

Certificate Management Profile

209 1 Scope

210 The *Certificate Management Profile* specializes the [Credential Management Profile](#) and extends the
211 management capability of the referencing profiles by adding the capability to model and manage X509
212 certificates of the public key infrastructure (PKI). This profile is not intended to serve as a mechanism for
213 the digital identification. Creation, storage, and management of X509 certificates, public private key pairs,
214 certificate revocation lists (CRL) and certificate signing requests (CSR) is detailed. Profile registration for
215 the schema implementation version information is also described.

216 2 Normative references

217 The following referenced documents are indispensable for the application of this document. For dated or
218 versioned references, only the edition cited (including any corrigenda or DMTF update versions) applies.
219 For references without a date or version, the latest published edition of the referenced document
220 (including any corrigenda or DMTF update versions) applies.

221 DMTF DSP0200, *CIM Operations over HTTP 1.3*,
222 http://www.dmtf.org/standards/published_documents/DSP0200_1.3.pdf

223 DMTF DSP0004, *CIM Infrastructure Specification 2.6*,
224 http://www.dmtf.org/standards/published_documents/DSP0004_2.6.pdf

225 DMTF DSP1001, *Management Profile Specification Usage Guide 1.0*,
226 http://www.dmtf.org/standards/published_documents/DSP1001_1.0.pdf

227 DMTF DSP1033, *Profile Registration Profile 1.0*,
228 http://www.dmtf.org/standards/published_documents/DSP1033_1.0.pdf

229 DMTF DSP1082, *Credential Management Profile 1.0*,
230 http://www.dmtf.org/standards/published_documents/DSP1082_1.0.pdf

231 IETF RFC 2986, *PKCS #10: Certification Request Syntax Specification, Version 1.7*, November 2000,
232 <http://www.ietf.org/rfc/rfc2986.txt>

233 IETF RFC3280, *Internet X.509 Public Key Infrastructure: Certificate and Certificate Revocation List (CRL)
234 Profile*, April 2002, <http://www.ietf.org/rfc/rfc3280.txt>

235 IETF RFC4514, *Lightweight Directory Access Protocol (LDAP): String Representation of Distinguished
236 Names*, June 2006, <http://www.ietf.org/rfc/rfc4514.txt>

237 IETF RFC5234, *Augmented BNF for Syntax Specifications: ABNF*, January 2008,
238 <http://www.ietf.org/rfc/rfc5234.txt>

239 ISO/IEC Directives, Part 2, *Rules for the structure and drafting of International Standards*,
240 <http://isotc.iso.org/livelink/livelink.exe?func=ll&objId=4230456&objAction=browse&sort=subtype>

241 3 Terms and definitions

242 In this document, some terms have a specific meaning beyond the normal English meaning. Those terms
243 are defined in this clause.

244 The terms "shall" ("required"), "shall not," "should" ("recommended"), "should not" ("not recommended"),
245 "may," "need not" ("not required"), "can" and "cannot" in this document are to be interpreted as described
246 in [ISO/IEC Directives, Part 2](#), Annex H. The terms in parenthesis are alternatives for the preceding term,
247 for use in exceptional cases when the preceding term cannot be used for linguistic reasons. Note that
248 [ISO/IEC Directives, Part 2](#), Annex H specifies additional alternatives. Occurrences of such additional
249 alternatives shall be interpreted in their normal English meaning.

250 The terms "clause," "subclause," "paragraph," and "annex" in this document are to be interpreted as
251 described in [ISO/IEC Directives, Part 2](#), Clause 5.

252 The terms "normative" and "informative" in this document are to be interpreted as described in [ISO/IEC](#)
253 [Directives, Part 2](#), Clause 3. In this document, clauses, subclauses, or annexes labeled "(informative)" do
254 not contain normative content. Notes and examples are always informative elements.

255 The terms defined in [DSP0004](#), [DSP0200](#), and [DSP1001](#) apply to this document. The following additional
256 terms are used in this document.

257 **3.1**

258 **Asymmetric Key Pair**

259 public/private key pair represented by CIM_UnsignedCredential

260 **3.2**

261 **Container Keystore**

262 the single instance of CIM_Keystore that contains a given PKI Credential Instance
263 The instance of CIM_Keystore is associated to the PKI Credential Instance through the
264 CIM_MemberOfCollection association.

265 **3.3**

266 **Managing Service**

267 The instance of CIM_CertificateManagementService that is managing a given PKI Credential Instance
268 The instance of CIM_CertificateManagementService is associated to the Container Keystore of the given
269 PKI Credential Instance through the CIM_ServiceAffectsElement association.

270 **3.4**

271 **PKI Credential Instance**

272 an instance of CIM_UnsignedCredential, CIM_X509Certificate, or CIM_X509CRL representing a PKI
273 credential

274 **4 Symbols and abbreviated terms**

275 The abbreviations defined in [DSP0004](#), [DSP0200](#), and [DSP1001](#) apply to this document. The following
276 additional abbreviations are used in this document.

277 **3.5**

278 **CRL**

279 Certification Revocation List

280 **3.6**

281 **CSR**

282 Certificate Signing Request

283 **3.7**

284 **PKI**

285 Public Key Infrastructure

286 5 Synopsis

287 **Profile Name:** Certificate Management

288 **Version:** 1.0.0

289 **Organization:** DMTF

290 **CIM schema version:** 2.29

291 **Central Class:** CIM_CertificateManagementService

292 **Scoping Class:** CIM_System

293 The *Certificate Management Profile* specializes the [Credential Management Profile](#) to extend the
294 management capability of the referencing profiles by adding the capability to represent and manage X509
295 certificates.

296 The Central Class of the *Certificate Management Profile* shall be CIM_CertificateManagementService.
297 The Central Instance shall be an instance of CIM_CertificateManagementService. The Scoping Class
298 shall be CIM_System. The Scoping Instance shall be the instance of CIM_System that is associated with
299 the Central Instance through the CIM_HostedService association.

300 Table 1 lists the profiles related to the *Certificate Management Profile*.

301 **Table 1 – Referenced profiles**

Profile Name	Organization	Version	Relationship	Behavior
Credential Management	DMTF	1.0	Specializes	
Profile Registration	DMTF	1.0	Mandatory	

302 6 Description

303 The *Certificate Management Profile* describes the properties and methods for X509 certificate
304 management in a managed system. This profile does not provide a mechanism for an application to
305 authenticate using certificates but rather to manage the certificates that are used by the managed system
306 for the authentication.

307 Figure 1 represents the class schema for the profile. For simplicity, the prefix *CIM_* has been removed
308 from the names of the classes.

- 327 1) A web service running on the managed system utilizes the owned key store for the X509
328 certificate that is presented to the connecting web client.
- 329 2) LDAP client services on the managed system utilize the trusted key store to store trusted X509
330 certificates to verify against the certificate presented by the LDAP server, which the managed
331 system connects to in the process of authentication.

332 Both the owned key store and the trusted key store are represented by the CIM_Keystore class which is
333 associated to the CIM_ManagedElement class representing the service that utilizes the credential store
334 by the CIM_ConcreteDependency association.

335 6.2 Public key infrastructure

336 Public key infrastructure is used by the managed system for establishing trust. The credentials used in
337 public key infrastructure are represented by the classes derived from the CIM_Credential class, such as
338 CIM_UnsignedCredential, CIM_X509Certificate, and CIM_X509CRL.

339 6.2.1 Asymmetric key pair

340 Asymmetric key pair on a managed system is represented by CIM_UnsignedCredential. The asymmetric
341 key pair can be used as an input for generating a Certificate Signing Request or a self-signed certificate
342 through the methods in the CIM_CertificateManagementService. CIM_CertificateManagementService
343 also contains an extrinsic method for importing an asymmetric key pair.

344 6.2.2 X.509 certificate

345 X.509 certificate is represented by the CIM_X509Certificate class. CIM_X509Certificate contains
346 properties corresponding to the X.509 certificate fields defined in [RFC3280](#).

347 If the X.509 certificate was generated using imported asymmetric key, then the CIM_UnsignedCredential
348 instance representing the asymmetric key is associated to the CIM_X509Certificate instance representing
349 the generated X.509 certificate through CIM_ConcreteDependency association, where the Antecedent
350 property references the CIM_UnsignedCredential instance and the Dependent property references the
351 CIM_X509Certificate instance.

352 If the managed element that uses X.509 certificate for authentication is represented by a subclass of
353 CIM_ManagedElement, this instance of the subclass of CIM_ManagedElement will be associated with the
354 CIM_X509Certificate instance using the CIM_CredentialContext association or will be associated with the
355 CIM_Keystore instance aggregating the CIM_X509Certificate instance using the
356 CIM_ConcreteDependency association. Thus, CIM_ConcreteDependency and CIM_CredentialContext
357 represent the usage of the credentials.

358 Certificate chains are also represented by the CIM_ConcreteDependency class associating the
359 CIM_X509Certificate instances representing the signer and signed certificates.
360 CIM_CertificateManagementService contains methods for creating, importing, and exporting certificates.

361 6.2.3 X.509 certificate revocation list (CRL)

362 X.509 CRL is represented by CIM_X509CRL class. CIM_X509CRL contains properties corresponding to
363 the X.509 CRL fields defined in [RFC3280](#). CIM_CertificateManagementService contains a method for
364 applying the encoded X.509 CRL received from the Certificate Authority onto a specific key store.

365 6.3 Authorization

366 X.509 certificates and keys may have different levels of access authorization. An authorized entity is
367 represented by a security principal through the CIM_Identity class. A security principal may be authorized
368 to access the credential store or a particular credential within a key store. The AssociatedPrivilege

369 association contains the privileges of the security principal as well as references to the key store, the
370 certificate/key, or both.

371 When an implementation has the ability to authorize on both levels (per key store and per the
372 certificate/key), the implementation calculates the effective authorization privileges for a particular security
373 principal by combining the key stores and its members' privileges in one of the following ways:

374 1) Collection Privileges Override – The effective privileges are the key store privileges overriding a
375 particular certificate/key's privileges.

376 2) Member Privileges Override – The effective credential privileges are the particular
377 certificate/key privileges overriding the key store's privileges.

378 3) Collection-Member Privileges Union – The effective credential privilege is the union of the key
379 store privileges and the particular certificate/key privileges.

380 4) Collection-Member Privileges Intersection – The effective credential privilege is the intersection
381 of the key store privileges and the particular certificate/key privileges.

382 The implementation supporting the key store level and member level privileges will implement one of the
383 above methodologies for calculating the effective privilege for a key/certificate. The
384 CertificateManagementCapabilities class will advertise which of these methodologies the implementation
385 supports.

386 7 Implementation

387 This clause details the requirements related to the arrangement of instances and their properties for
388 implementations of this profile.

389 7.1 Key store

390 The requirements for the instances of CIM_Keystore and PKI Credential Instances are defined in the
391 [Credential Management Profile](#).

392 7.2 Certificate management service

393 This subclause details the requirements for the certificate management service and the representation of
394 its capabilities.

395 7.2.1 CIM_CertificateManagementService

396 Each PKI-based credential represented by a PKI Credential Instance shall be managed by one and only
397 one certificate management service represented by an instance of CIM_CertificateManagementService,
398 also referred to as a Managing Service (see 3.3). Each Container Keystore shall be associated to an
399 instance of CIM_CertificateManagementService. The PKI Credential Instances within the Container
400 Credential Store shall be managed by the CIM_CertificateManagementService instance, which the
401 Container Keystore is associated through CIM_ServiceAffectsElement.

402 7.2.2 CIM_CertificateManagementCapabilities

403 There shall be one and only one instance of CIM_CertificateMangementCapabilities associated with an
404 instance of CIM_CertificateManagementService representing the capabilities of a certificate management
405 service.

406 If the AsymmetricKeyGeneration property is TRUE, then the KeyAlgorithmSupportedProperty shall have a
407 non empty, non-NULL value.

408 If the SupportedMethods array property contains a value of 103 (ImportEncodedCertificates) or 106
 409 (ApplyCRL), then the InputFormatsSupported array property shall have a non empty, non-NULL value.

410 If the SupportedMethods array property contains a value of 105 (ExportEncodedCertificate), then the
 411 OutputFormatsSupported array property shall have a non empty, non-NULL value.

412 **7.3 Certificate chain**

413 Certificate chains may be represented. If the complete chain of certificates is implemented, the
 414 requirements in this subclause shall apply.

415 Each certificate in the certificate chain shall be represented by an instance of CIM_X509Certificate. The
 416 instance of CIM_X509Certificate representing the signing certificate shall be associated with instances of
 417 CIM_X509Certificate that represent the signed certificates using CIM_ConcreteDependency, where the
 418 Antecedent property shall reference the instance of CIM_X509Certificate representing the signing
 419 certificate and the Dependent property shall reference the instance of CIM_X509Certificate representing
 420 the signed certificate.

421 **7.4 Authorization**

422 This subclause details the requirements that are in addition to the authorization requirements in the
 423 [Credential Management Profile](#).

424 If the security principal authorization is implemented, then the CIM_Identity instance representing the
 425 security principal shall be associated with a CIM_KeyStore instance or PKI Credential Instance through
 426 the CIM_AssociatedPrivilege association.

427 For the CIM_Keystore instance referenced by CIM_AssociatedPrivilege, the referenced CIM_Identity shall
 428 be authorized to perform the intrinsic operations or execute extrinsic methods in the “Intrinsic Operations”
 429 and “Extrinsic Methods” columns of Table 2, if and only if the CIM_AssociatedPrivilege.Activities property
 430 contains the value from the “CIM_AssociatedPrivilege.Activities” column of the respective row of Table 2.

431 **Table 2 – CIM_AssociatedPrivilege.Activities mapping to PKI Credential Instance operations**

CIM_AssociatedPrivilege.Activities	Credential Operation	Intrinsic Operations	Extrinsic Methods
2 (Create) or 6 (Write)	Import a key/certificate into a key store	Modify Instance on PKI Credential Instance	ImportPublicPrivateKeyPair() CreateCertificateSigningRequest() CreateSelfSignedCertificate() ImportEncodedCertificates() ImportCertificates() ApplyCRL() ApplyDecodedCRL()
5 (Read)	Export a key/certificate from a key store	Get on PKI Credential Instance Enumerate on PKI Credential Instance	ExportEncodedCertificates()
3 (Delete)	Delete a key/certificate from a key store	Delete on PKI Credential Instance	

432 For the PKI Credential Instance referenced by CIM_AssociatedPrivilege, the referenced CIM_Identity
 433 shall be authorized to perform the intrinsic operations or execute extrinsic methods in the “Intrinsic
 434 Operations” and “Extrinsic Methods” columns of Table 3, if and only if the
 435 CIM_AssociatedPrivilege.Activities property contains the value from “CIM_AssociatedPrivilege.Activities”
 436 column of the respective row of Table 3.

437 **Table 3 – CIM_AssociatedPrivilege.Activities mapping to PKI Credential Instance operations**

CIM_AssociatedPrivilege.Activities	Credential Operation	Intrinsic Operations	Extrinsic Methods
6 (Write)	Modify a key or certificate	Modify Instance on PKI Credential Instance	
5 (Read)	Get a key or certificate	Get on PKI Credential Instance Enumerate on PKI Credential Instance	ExportEncodedCertificates()
3 (Delete)	Delete a key or certificate	Delete on PKI Credential Instance	

438 8 Methods

439 This clause details the requirements for supporting intrinsic operations and extrinsic methods for the CIM
 440 elements defined by this profile.

441 8.1 CIM_CertificateManagementService.ImportPublicPrivateKeyPair()

442 The ImportPublicPrivateKeyPair() method provides the ability to import a public/private key pair to be
 443 used by the managed system.

444 The ImportPublicPrivateKeyPair() method’s return code values shall be as specified in Table 4, where the
 445 method execution behavior matches the return code description. The ImportPublicPrivateKeyPair()
 446 method’s parameters are specified in

447

448

449 Table 5.

450 No standard messages are defined for this method.

451 **Table 4 – CIM_CertificateManagementService.ImportPublicPrivateKeyPair() method: Return code**
 452 **values**

Value	Description
0	Operation completed successfully
1	Operation unsupported
2	Failed
4096	Job created – only Job output parameter shall be returned.

453

454

455

456 **Table 5 – CIM_CertificateManagementService.ImportPublicPrivateKeyPair() method: Parameters**

Qualifiers	Name	Type	Description/Values
IN, REQ, OctetString	PublicKey[]	string	Public key as an octet string array of one element. The public key shall be contained in the first element of the array. All other array element shall be ignored.
IN, REQ, OctetString	PrivateKey[]	string	Private key as an octet string array of one element. The private key shall be contained in the first element of the array. All other array element shall be ignored.
IN, REQ	Keystore	CIM_Keystore REF	References the key store to which the asymmetric key is to be added
IN	CredentialContext	CIM_ManagedElement REF	References an instance of CIM_ManagedElement that will utilize the imported credential
IN	Usage[]	uint16	Describes how the managed element, referenced by the CredentialContext parameter, utilizes the imported credential
OUT	Job	CIM_ConcreteJob REF	Reference to the ConcreteJob created to track the execution initiated by the method invocation.
OUT, REQ	PPKPCredential	CIM_UnsignedCredential REF	References the instance representing the imported public/private key pair.

457 **8.1.1 CIM_CertificateManagementService.ImportPublicPrivateKeyPair() conditional**
 458 **support**

459 If the SupportedMethods property array of the associated instance of
 460 CIM_CertificateManagementCapabilities contains the value 2 (ImportPublicPrivateKeyPair), the
 461 ImportPublicPrivateKeyPair() method shall be implemented and shall not return the value 1 (Not
 462 Supported).

463 If the SupportedMethods property array of the associated instance of
 464 CIM_CertificateManagementCapabilities does not contain the value 2 (ImportPublicPrivateKeyPair), the
 465 ImportPublicPrivateKeyPair() method shall not be implemented or shall always return the value 1 (Not
 466 Supported).

467 **8.1.2 Parameter validation**

468 If the Usage parameter has a non-NULL value, then the CredentialContext parameter shall have a non-
 469 NULL value. If the Usage parameter has non-NULL value and the CredentialContext parameter is NULL,
 470 then the method shall return 2 (Failed).

471 **8.1.3 Required privileges**

472 The method invocation shall fail if the Activities property of the CIM_AssociatedPrivilege instance,
 473 referencing the CIM_Identity instance representing the executor's security principal, does not contain the
 474 value 2 (Create) or 6 (Write).

475 **8.2 CIM_CertificateManagementService.CreateKeystore()**

476 The CreateKeystore() method provides the ability to create new key stores.

477 The CreateKeystore() method's return code values shall be as specified in Table 6 where the method
 478 execution behavior matches the return code description. The CreateKeystore() method's parameters are
 479 specified in Table 7.

480 No standard messages are defined for this method.

481 **Table 6 – CIM_CertificateManagementService.CreateKeystore() method: Return code values**

Value	Description
0	Operation completed successfully
1	Operation unsupported
2	Failed
4096	Job created – only Job output parameter shall be returned.

482 **Table 7 – CIM_CertificateManagementService.CreateKeystore() method: Parameters**

Qualifiers	Name	Type	Description/Values
IN, REQ	Keystore	String	Embedded instance representing the new key store
IN, REQ	OwningSystem	CIM_System REF	Reference to the owning managed system for the key store
IN	KeystoreUtilizers[]	CIM_ManagedElement REF	Array of references to the CIM_ManagedElement subclass instances that will utilize the new key store
OUT	Job	CIM_ConcreteJob REF	Reference to the ConcreteJob created to track the execution initiated by the method invocation
OUT, REQ	Keystore	CIM_Keystore REF	Reference to the newly created CIM_Keystore instance representing the new key store

483 **8.2.1 CIM_CertificateManagementService.CreateKeystore() conditional support**

484 If the SupportedMethods property array of the associated instance of
 485 CIM_CertificateManagementCapabilities contains the value 3 (CreateKeystore), the CreateKeystore()
 486 method shall be implemented and shall not return the value 1 (Not Supported).

487 If the SupportedMethods property array of the associated instance of
 488 CIM_CertificateManagementCapabilities does not contain the value 3 (CreateKeystore), the
 489 CreateKeystore() method shall not be implemented or shall always return the value 1 (Not Supported).

490 **8.2.2 Parameter validation**

491 The Keystore parameter shall have non-NULL values for the Usage property of the CIM_Keystore
 492 embedded instance. If the Usage property of the CIM_Keystore embedded instance of the Keystore
 493 parameter is NULL or is missing, the method shall return a value of 2 (Failed).

494 **8.3 CIM_CertificateManagementService.CreateCertificateSigningRequest()**

495 The CreateCertificateSigningRequest() method provides the ability to create CSRs based on PKCS#10
 496 referenced in [RFC2986](#).

497 The CreateCertificateSigningRequest() method's return code values shall be as specified in Table 8,
 498 where the method execution behavior matches the return code description. The
 499 CreateCertificateSigningRequest() method's parameters are specified in Table 9.

500 No standard messages are defined for this method.

501 **Table 8 – CIM_CertificateManagementService.CreateCertificateSigningRequest() method: Return**
 502 **code values**

Value	Description
0	Operation completed successfully
1	Operation unsupported
2	Failed
4096	Job created – only Job output parameter shall be returned.

503 **Table 9 – CIM_CertificateManagementService.CreateCertificateSigningRequest() method:**
 504 **Parameters**

Qualifiers	Name	Type	Description/Values
IN	Subject	string	Subject shall contain information as required by section 4.1.2.4 of RFC3280 , formatted based on RFC4514 .
IN	AltSubject	string	String containing alternate subject identifier
IN	PublicKeyAlgorithm	uint16	The algorithm used for generating the public/private key pair. This parameter shall be supported if the AsymmetricKeyGeneration property of the associated CIM_CertificateManagementCapabilities instance is TRUE.
IN	PublicKeySize	uint16	The length of the public key. This parameter shall be supported if the AsymmetricKeyGeneration property of the associated CIM_CertificateManagementCapabilities is TRUE and if the PublicPrivateKeyPair is not specified.
IN	PublicPrivateKeyPair	CIM_UnsignedCredential REF	Reference to the instance of CIM_UnsignedCredential representing the public/private key pair

Qualifiers	Name	Type	Description/Values
IN	ExtendedKeyUsageValue[]	string	Indicates one or more purposes for which the certified public key may be used. The value at each index describes the key usage value based on the type at the corresponding index of ExtendedKeyUsageType[] parameter array.
IN	ExtendedKeyUsageType[]	uint16	Indicates the type for ExtendedKeyUsageValue based on the ASN.1 GeneralName types. The value at each index describes the type for ExtendedKeyUsageValue parameter at the corresponding index.
IN	SignatureAlgorithm	uint16	Indicates the signature algorithm used to sign the CSR
IN, REQ	OutputFormat	uint16	Indicates the output format of the CSR
OUT	Job	CIM_ConcreteJob REF	Reference to the ConcreteJob created to track the execution initiated by the method invocation.
OUT, REQ, OctetString	CSR[]	string	CSR as a single-element octet string array. The entire CSR shall be contained in the first array element. All other array elements shall be ignored, if present.

505 **8.3.1 CIM_CertificateManagementService.CreateCertificateSigningRequest() conditional**
506 **support**

507 If the SupportedMethods property array of the associated instance of
508 CIM_CertificateManagementCapabilities contains the value 101 (CreateCertificateSigningRequest), the
509 CreateCertificateSigningRequest() method shall be implemented and shall not return the value 1 (Not
510 Supported).

511 If the SupportedMethods property array of the associated instance of
512 CIM_CertificateManagementCapabilities does not contain the value
513 101 (CreateCertificateSigningRequest), the CreateCertificateSigningRequest() method shall not be
514 implemented or shall always return the value 1 (Not Supported).

515 **8.3.2 Parameter validation**

516 If the PublicPrivateKeyPair parameter is NULL and the AsymmetricKeyGeneration property of the
517 associated instance of CIM_CertificateManagementCapabilities has the value FALSE, then the method
518 shall return a value of 2 (Failed).

519 If the PublicPrivateKeyPair parameter is NULL and the PublicKeyAlgorithm or PublicKeySize parameters
520 are NULL, then the method shall return a value of 2 (Failed).

521 If the PublicKeyAlgorithm parameter is not NULL and the KeyAlgorithmSupported array property of the
522 associated instance of CIM_CertificateManagementCapabilities does not contain the value of the
523 PublicKeyAlgorithm parameter, then the method shall return a value of 2 (Failed).

524 If the Subject and AltSubject parameters are both NULL, then the method shall return a value of 2
525 (Failed).

526 If the OutputFormat parameter does not contain a value specified in the SupportedOutputFormats
 527 property array of the associated CIM_CertificateManagementCapabilities instance, then the method shall
 528 return a value 2 (Failed).

529 If the CIM_CertificateManagementCapabilities.SupportedSignatureAlgorithms property is implemented
 530 and the SignatureAlgorithm parameter does not contain a value specified in the
 531 SupportedSignatureAlgorithms property array, then the method shall return a value 2 (Failed).

532 **8.3.3 Required privileges**

533 The method invocation shall fail if the Activities property of the CIM_AssociatedPrivilege instance,
 534 referencing the CIM_Identity instance representing the executor's security principal, does not contain a
 535 value of 2 (Create) or 6 (Write).

536 **8.4 CIM_CertificateManagementService.CreateSelfSignedCertificate()**

537 The CreateSelfSignedCertificate() method provides the ability to create an X509 self-signed certificate
 538 either based on the pre-generated public-private key pair or based on the method execution generated
 539 public private key-pair.

540 The CreateSelfSignedCertificate() method's return code values shall be as specified in Table 10 where
 541 the method execution behavior matches the return code description. The CreateSelfSignedCertificate()
 542 method's parameters are specified in Table 11.

543 If the pre-generated public-private key pair is used by providing a valid PublicPrivateKeyPair parameter,
 544 then the generated X.509 certificate in the SelfSignedCertificate output parameter shall reference the
 545 CIM_UnsignedCredential instance representing the PublicPrivateKeyPair through
 546 CIM_ConcreteDependency association, where the Antecedent property references the
 547 CIM_UnsignedCredential instance and the Dependent property references the CIM_X509Certificate
 548 instance.

549 No standard messages are defined for this method.

550 **Table 10 – CIM_CertificateManagementService.CreateSelfSignedCertificate() method: Return code**
 551 **values**

Value	Description
0	Operation completed successfully
1	Operation unsupported
2	Failed
4096	Job created – only Job output parameter shall be returned.

552 **Table 11 – CIM_CertificateManagementService.CreateSelfSignedCertificate() method: Parameters**

Qualifiers	Name	Type	Description/Values
IN	Subject	string	Subject shall contain information as required by section 4.1.2.4 of RFC3280 , formatted based on RFC4514
IN	AltSubject	string	String containing alternate subject identifier

Qualifiers	Name	Type	Description/Values
IN	PublicKeyAlgorithm	uint16	The algorithm used for generating public/private key pair. This parameter shall be supported if the AsymmetricKeyGeneration property of the associated CIM_CertificateManagementCapabilities is TRUE and if the PublicPrivateKeyPair is not specified.
IN	PublicKeySize	uint16	The length of the public key. This parameter shall be supported if the AsymmetricKeyGeneration property of the associated CIM_CertificateManagementCapabilities is TRUE.
IN	PublicPrivateKeyPair	CIM_UnsignedCredential REF	Reference to the instance of CIM_UnsignedCredential representing the public/private key pair
IN	Keystore	CIM_Keystore REF	References the key store to add the newly created X509 self-signed certificate
IN	CredentialContext	CIM_ManagedElement REF	References an instance of CIM_ManagedElement that will utilize the newly created X509 self-signed certificate
IN	Usage	uint16	Describes how the managed element, referenced by CredentialContext parameter, utilizes the self-signed certificate
IN	SignatureAlgorithm	uint16	Indicates the signature algorithm used to sign the X509 self-signed certificate
OUT	Job	CIM_ConcreteJob REF	Reference to the ConcreteJob instance created to track the execution initiated by the method invocation
OUT, REQ	SelfSignedCertificate	CIM_X509Certificate REF	References the newly created X509 self-signed certificate

553 8.4.1 CIM_CertificateManagementService.CreateSelfSignedCertificate() conditional 554 support

555 If the SupportedMethods property array of the associated instance of
556 CIM_CertificateManagementCapabilities contains the value 102 (CreateSelfSignedCertificate), the
557 CreateSelfSignedCertificate() method shall be implemented and shall not return the value 1 (Not
558 Supported).

559 If the SupportedMethods property array of the associated instance of
560 CIM_CertificateManagementCapabilities does not contain the value 102 (CreateSelfSignedCertificate),
561 the CreateSelfSignedCertificate() method shall not be implemented or shall always return the value
562 1 (Not Supported).

563 **8.4.2 Parameter validation**

564 If the `PublicPrivateKeyPair` parameter is NULL and the `AsymmetricKeyGeneration` property of the
 565 associated instance of `CIM_CertificateManagementCapabilities` has the value FALSE, then the method
 566 shall return a value of 2 (Failed).

567 If the `PublicPrivateKeyPair` parameter is NULL and the `PublicKeyAlgorithm` or `PublicKeySize` parameters
 568 are NULL, then the method shall return a value of 2 (Failed).

569 If the `PublicKeyAlgorithm` parameter is not NULL and the `KeyAlgorithmSupported` array property of the
 570 associated instance of `CIM_CertificateManagementCapabilities` does not contain the value of the
 571 `PublicKeyAlgorithm` parameter, then the method shall return a value of 2 (Failed).

572 If the `Subject` and `AltSubject` parameters are both NULL, the method shall return a value of 2 (Failed).

573 If the `CIM_CertificateManagementCapabilities.SupportedSignatureAlgorithms` property is implemented
 574 and the `SignatureAlgorithm` parameter does not contain a value specified in the
 575 `SupportedSignatureAlgorithms` property array, then the method shall return a value 2 (Failed).

576 **8.4.3 Required privileges**

577 The method invocation shall fail if the `Activities` property of the `CIM_AssociatedPrivilege`, referencing the
 578 `CIM_Identity` instance representing the executor's security principal, does not contain a value of 2
 579 (Create) or 6 (Write).

580

581 **8.5 CIM_CertificateManagementService.ImportEncodedCertificates()**

582 The `ImportEncodedCertificates()` method provides the ability to import encoded X509 certificates.

583 The `ImportEncodedCertificates()` method's return code values shall be as specified in Table 12, where
 584 the method execution behavior matches the return code description. The `ImportEncodedCertificates()`
 585 method's parameters are specified in Table 13.

586 No standard messages are defined for this method.

587 **Table 12 – CIM_CertificateManagementService.ImportEncodedCertificates() method: Return code**
 588 **values**

Value	Description
0	Operation completed successfully
1	Operation unsupported
2	Failed
4096	Job created – only Job output parameter shall be returned.

589 **Table 13 – CIM_CertificateManagementService.ImportEncodedCertificates() method: Parameters**

Qualifiers	Name	Type	Description/Values
IN, REQ	<code>EncodedCertificates[]</code>	string	Single-element octet string array representing the encoded X509 certificates
IN, REQ	<code>Format</code>	uint16	Specifies the format for the encoding of the encoded X509 certificates

Qualifiers	Name	Type	Description/Values
IN	Keystore	CIM_Keystore REF	References the key store to which the newly imported X509 certificates are to be added
IN	CredentialContext	CIM_ManagedElement REF	References an instance of CIM_ManagedElement that will utilize the newly imported X509 certificates
IN	Usage[]	uint16	Describes how the managed element, referenced by CredentialContext parameter, utilizes the imported certificates
OUT	Job	CIM_ConcreteJob REF	Reference to the ConcreteJob created to track the execution initiated by the method invocation
OUT, REQ	NewCertificates[]	CIM_X509Certificate REF	References newly imported X509 certificates

590 **8.5.1 CIM_CertificateManagementService.ImportEncodedCertificates() conditional** 591 **support**

592 If the SupportedMethods property array of the associated instance of
593 CIM_CertificateManagementCapabilities contains the value 103 (ImportEncodedCertificates), the
594 ImportEncodedCertificates() method shall be implemented and shall not return the value 1 (Not
595 Supported).

596 If the SupportedMethods property array of the associated instance of
597 CIM_CertificateManagementCapabilities does not contain the value 103 (ImportEncodedCertificates), the
598 ImportEncodedCertificates() method shall not be implemented or shall always return the value 1 (Not
599 Supported).

600 **8.5.2 Parameter validation**

601 If the Format parameter does not contain a value from the InputFormatsSupported array property of the
602 associated instance of CIM_CertificateManagementCapabilities, then the method shall return a value of 2
603 (Failed).

604 **8.5.3 Required privileges**

605 The method invocation shall fail if the Activities property of the CIM_AssociatedPrivilege, referencing the
606 CIM_Identity instance representing the executor's security principal, does not contain 2 (Create) or 6
607 (Write) values.

608 **8.6 CIM_CertificateManagementService.ImportCertificates()**

609 The ImportCertificates() method provides the ability to import X509 certificates as embedded instances of
610 the CIM_X509Certificate class.

611 The ImportCertificates() method's return code values shall be as specified in Table 14, where the method
612 execution behavior matches the return code description. The ImportCertificates() method's parameters
613 are specified in Table 15.

614 No standard messages are defined for this method.

615 **Table 14 – CIM_CertificateManagementService.ImportCertificates() method: Return code values**

Value	Description
0	Operation completed successfully
1	Operation unsupported
2	Failed
4096	Job created – only Job output parameter shall be returned.

616 **Table 15 – CIM_CertificateManagementService.ImportCertificates() method: Parameters**

Qualifiers	Name	Type	Description/Values
IN, REQ	InputCertificates[]	string	Embedded instances of CIM_X509Certificate representing the imported certificates
IN	Keystore	CIM_Keystore REF	References the key store to which to add the newly imported X509 certificates
IN	CredentialContext	CIM_ManagedElement REF	References an instance of CIM_ManagedElement that will utilize the newly imported X509 certificates
IN	Usage[]	uint16	Describes how the managed element, referenced by the CredentialContext parameter, utilizes the imported certificates
OUT	Job	CIM_ConcreteJob REF	Reference to the ConcreteJob instance created to track the execution initiated by the method invocation
OUT, REQ	NewCertificates[]	CIM_X509Certificate REF	References newly imported X509 certificates

617 **8.6.1 CIM_CertificateManagementService.ImportCertificates() conditional support**

618 If the SupportedMethods property array of the associated instance of
 619 CIM_CertificateManagementCapabilities contains the value 104 (ImportCertificates), the
 620 ImportCertificates() method shall be implemented and shall not return the value 1 (Not Supported).

621 If the SupportedMethods property array of the associated instance of
 622 CIM_CertificateManagementCapabilities does not contain the value 104 (ImportCertificates), the
 623 ImportCertificates() method shall not be implemented or shall always return the value 1 (Not Supported).

624 **8.6.2 Required privileges**

625 The method invocation shall fail if the Activities property of the CIM_AssociatedPrivilege, referencing the
 626 CIM_Identity instance representing the executor’s security principal, does not contain a value of 2
 627 (Create) or 6 (Write).

628 **8.7 CIM_CertificateManagementService.ExportEncodedCertificates()**

629 The ExportEncodedCertificates() method provides the ability to export X509 certificates using a specified
 630 encoded format.

631 The ExportEncodedCertificates() method's return code values shall be as specified in Table 16 where the
 632 method execution behavior matches the return code description. The ExportEncodedCertificates()
 633 method's parameters are specified in Table 17.

634 No standard messages are defined for this method.

635 **Table 16 – CIM_CertificateManagementService.ExportEncodedCertificates() method: Return code**
 636 **values**

Value	Description
0	Operation completed successfully
1	Operation unsupported
2	Failed
4096	Job created – only Job output parameter shall be returned.

637 **Table 17 – CIM_CertificateManagementService.ExportEncodedCertificates() method: Parameters**

Qualifiers	Name	Type	Description/Values
IN, REQ	CertificatesToExport[]	CIM_X509Certificate REF	Reference to the CIM_ManagedElement for which metrics will be controlled
IN, REQ	Format	uint16	Specifies the format for the encoding of the encoded X509 certificates to export
OUT	Job	CIM_ConcreteJob REF	Reference to the ConcreteJob created to track the execution initiated by the method invocation
OUT, REQ, OctetString	EncodedCertificates[]	string	Single-element octet string array representing the exported X509 certificates. All the X509 certificates shall be contained in the first element of the array. All other array element shall be ignored.

638 **8.7.1 CIM_CertificateManagementService.ExportEncodedCertificates() conditional** 639 **support**

640 If the SupportedMethods property array of the associated instance of
 641 CIM_CertificateManagementCapabilities contains the value 105 (ExportEncodedCertificate), the
 642 ExportEncodedCertificates() method shall be implemented and shall not return the value 1 (Not
 643 Supported).

644 If the SupportedMethods property array of the associated instance of
 645 CIM_CertificateManagementCapabilities does not contain the value 105 (ExportEncodedCertificate), the
 646 ExportEncodedCertificates() method shall not be implemented or shall always return the value 1 (Not
 647 Supported).

648 **8.7.2 Parameter validation**

649 If the Format parameter does not contain a value from the OutputFormatsSupported array property of the
 650 associated instance of CIM_CertificateManagementCapabilities, then the method shall return a value of 2
 651 (Failed).

652 If the instances of CIM_X509Certificate in the adjacent indexes of the CertificatesToExport array
 653 parameter are not associated with each other using the CIM_ConcreteDependency association, where

654 the instance at the lower index is referenced by the Antecedent property and the instance at the higher
 655 index is referenced by the Dependent property, the method shall return a value of 2 (Failed).

656 **8.7.3 Required privileges**

657 The method invocation shall fail if the Activities property of the CIM_AssociatedPrivilege instance,
 658 referencing the CIM_Identity instance representing the executor's security principal, does not contain a
 659 value of 5 (Read).

660 **8.8 CIM_CertificateManagementService.ApplyCRL()**

661 The ApplyCRL() method provides the ability to apply CRL in an encoded format.

662 The ApplyCRL() method's return code values shall be as specified in Table 18, where the method
 663 execution behavior matches the return code description. The ApplyCRL() method's parameters are
 664 specified in Table 19.

665 No standard messages are defined for this method.

666 **Table 18 – CIM_CertificateManagementService.ApplyCRL() method: Return code values**

Value	Description
0	Operation completed successfully
1	Operation unsupported
2	Failed
4096	Job created – only Job output parameter shall be returned.

667 **Table 19 – CIM_CertificateManagementService.ApplyCRL() method: Parameters**

Qualifiers	Name	Type	Description/Values
IN, REQ, OctetString	EncodedCRL[]	string	Single-element string octet array representing the encoded CRL. The encoded CRL shall be contained in the first element of the array. All other array element shall be ignored.
IN, REQ	Format	uint16	Specifies the format for the encoding of the encoded CRL
IN	Keystore	CIM_Keystore REF	References the key store to which the CRL will be applied
IN	CredentialContext	CIM_ManagedElement REF	References an instance of CIM_ManagedElement for which the certificates are revoked
OUT	Job	CIM_ConcreteJob REF	Reference to the ConcreteJob created to track the execution initiated by the method invocation
OUT, REQ	AppliedCRL	CIM_X509CRL REF	References the newly applied CRL

668 8.8.1 CIM_CertificateManagementService.ApplyCRL() conditional support

669 If the SupportedMethods property array of the associated instance of
670 CIM_CertificateManagementCapabilities contains the value 106 (ApplyCRL), the ApplyCRL() method
671 shall be implemented and shall not return the value 1 (Not Supported).

672 If the SupportedMethods property array of the associated instance of
673 CIM_CertificateManagementCapabilities does not contain the value 106 (ApplyCRL), the ApplyCRL()
674 method shall not be implemented or shall always return the value 1 (Not Supported).

675 8.8.2 Parameter validation

676 If the InputFormatsSupported array property of the associated instance of
677 CIM_CertificateManagementCapabilities does not contain the value of the Format parameter, the method
678 shall return a value of 2 (Failed).

679 8.8.3 Required privileges

680 The method invocation shall fail if the Activities property of the CIM_AssociatedPrivilege, referencing the
681 CIM_Identity instance representing the executor's security principal, does not contain a value of 2
682 (Create) or 6 (Write).

683 8.9 CIM_CertificateManagementService.ApplyDecodedCRL()

684 The ApplyDecodedCRL() method provides the ability to apply CRL in the decoded format.

685 The ApplyDecodedCRL() method's return code values shall be as specified in Table 18, where the
686 method execution behavior matches the return code description. The ApplyDecodedCRL() method's
687 parameters are specified in Table 19.

688 No standard messages are defined for this method.

689 Table 20 – CIM_CertificateManagementService.ApplyDecodedCRL() method: Return code values

Value	Description
0	Operation completed successfully
1	Operation unsupported
2	Failed
4096	Job created – only Job output parameter shall be returned.

690 Table 21 – CIM_CertificateManagementService.ApplyDecodedCRL() method: Parameters

Qualifiers	Name	Type	Description/Values
IN, REQ	Issuer	string	Issuer shall contain information as required by section 4.1.2.4 of RFC3280 , formatted based on RFC4514 .
IN, REQ	SerialNumbers[]	String	Serial numbers of the certificates contained in the CRL
IN	Keystore	CIM_Keystore REF	References the key store to which the CRL will be applied

Qualifiers	Name	Type	Description/Values
IN	CredentialContext	CIM_ManagedElement REF	References an instance of CIM_ManagedElement for which the certificates are revoked
OUT	Job	CIM_ConcreteJob REF	Reference to the ConcreteJob created to track the execution initiated by the method invocation
OUT, REQ	AppliedCRL	CIM_X509CRL REF	References newly applied CRL

691 **8.9.1 CIM_CertificateManagementService.ApplyDecodedCRL() conditional support**

692 If the SupportedMethods property array of the associated instance of
 693 CIM_CertificateManagementCapabilities contains the value 107 (ApplyDecodedCRL), the
 694 ApplyDecodedCRL() method shall be implemented and shall not return the value 1 (Not Supported).

695 If the SupportedMethods property array of the associated instance of
 696 CIM_CertificateManagementCapabilities does not contain the value 107 (ApplyDecodedCRL), the
 697 ApplyDecodedCRL() method shall not be implemented or shall always return the value 1 (Not
 698 Supported).

699 **8.9.2 Required privileges**

700 The method invocation shall fail if the Activities property of the CIM_AssociatedPrivilege instance,
 701 referencing the CIM_Identity instance representing the executor's security principal, does not contain a
 702 value of 2 (Create) or 6 (Write).

703 **8.10 Profile conventions for operations**

704 For each profile class (including associations), the implementation requirements for operations, including
 705 those in the following default list, are specified in class-specific subclauses of this clause.

706 The default list of operations is as follows:

- 707 • GetInstance
- 708 • Associators
- 709 • AssociatorNames
- 710 • References
- 711 • ReferenceNames
- 712 • EnumerateInstances
- 713 • EnumerateInstanceNames

714 **8.11 CIM_CertificateManagementCapabilities**

715 All operations in the default list in 8.10 shall be implemented as defined in [DSP0200](#).

716 NOTE: Related profiles may define additional requirements on operations for the profile class.

717 **8.12 CIM_CertificateManagementService**

718 All operations in the default list in 8.10 shall be implemented as defined in [DSP0200](#).

719 NOTE: Related profiles may define additional requirements on operations for the profile class.

720 **8.13 CIM_CredentialContext**

721 Table 22 lists implementation requirements for operations. If implemented, these operations shall be
 722 implemented as defined in [DSP0200](#). In addition, and unless otherwise stated in Table 22, all operations
 723 in the default list in 8.10 shall be implemented as defined in [DSP0200](#).

724 NOTE: Related profiles may define additional requirements on operations for the profile class.

725 **Table 22 – Operations: CIM_CredentialContext**

Operation	Requirement	Messages
Associators	Unspecified	None
AssociatorNames	Unspecified	None
References	Unspecified	None
ReferenceNames	Unspecified	None

726 **8.14 CIM_ConcreteDependency (CIM_Keystore)**

727 Table 23 lists implementation requirements for operations. If implemented, these operations shall be
 728 implemented as defined in [DSP0200](#). In addition, and unless otherwise stated in Table 23, all operations
 729 in the default list in 8.10 shall be implemented as defined in [DSP0200](#).

730 NOTE: Related profiles may define additional requirements on operations for the profile class.

731 **Table 23 – Operations: CIM_ConcreteDependency**

Operation	Requirement	Messages
Associators	Unspecified	None
AssociatorNames	Unspecified	None
References	Unspecified	None
ReferenceNames	Unspecified	None

732 **8.15 CIM_ConcreteDependency (CIM_X509Certificate)**

733 Table 24 lists implementation requirements for operations. If implemented, these operations shall be
 734 implemented as defined in [DSP0200](#). In addition, and unless otherwise stated in Table 24, all operations
 735 in the default list in 8.10 shall be implemented as defined in [DSP0200](#).

736 NOTE: Related profiles may define additional requirements on operations for the profile class.

737 **Table 24 – Operations: CIM_ConcreteDependency**

Operation	Requirement	Messages
Associators	Unspecified	None
AssociatorNames	Unspecified	None
References	Unspecified	None
ReferenceNames	Unspecified	None

738 **8.16 CIM_ElementCapabilities**

739 Table 25 lists implementation requirements for operations. If implemented, these operations shall be
 740 implemented as defined in [DSP0200](#). In addition, and unless otherwise stated in Table 25, all operations
 741 in the default list in 8.10 shall be implemented as defined in [DSP0200](#).

742 NOTE: Related profiles may define additional requirements on operations for the profile class.

743 **Table 25 – Operations: CIM_ElementCapabilities**

Operation	Requirement	Messages
Associators	Unspecified	None
AssociatorNames	Unspecified	None
References	Unspecified	None
ReferenceNames	Unspecified	None

744 **8.17 CIM_HostedService**

745 Table 26 lists implementation requirements for operations. If implemented, these operations shall be
 746 implemented as defined in [DSP0200](#). In addition, and unless otherwise stated in Table 26, all operations
 747 in the default list in 8.10 shall be implemented as defined in [DSP0200](#).

748 NOTE: Related profiles may define additional requirements on operations for the profile class.

749 **Table 26 – Operations: CIM_HostedService**

Operation	Requirement	Messages
Associators	Unspecified	None
AssociatorNames	Unspecified	None
References	Unspecified	None
ReferenceNames	Unspecified	None

750 **8.18 CIM_Keystore**

751 All operations in the default list in 8.10 shall be implemented as defined in [DSP0200](#).

752 NOTE: Related profiles may define additional requirements on operations for the profile class.

753 **8.19 CIM_MemberOfCollection**

754 Table 27 lists implementation requirements for operations. If implemented, these operations shall be
 755 implemented as defined in [DSP0200](#). In addition, and unless otherwise stated in Table 27, all operations
 756 in the default list in 8.10 shall be implemented as defined in [DSP0200](#).

757 NOTE: Related profiles may define additional requirements on operations for the profile class.

758 **Table 27 – Operations: CIM_MemberOfCollection**

Operation	Requirement	Messages
Associators	Unspecified	None
AssociatorNames	Unspecified	None
References	Unspecified	None
ReferenceNames	Unspecified	None

759 8.20 CIM_OwningCollectionElement

760 Table 28 lists implementation requirements for operations. If implemented, these operations shall be
 761 implemented as defined in [DSP0200](#). In addition, and unless otherwise stated in Table 28, all operations
 762 in the default list in 8.10 shall be implemented as defined in [DSP0200](#).

763 NOTE: Related profiles may define additional requirements on operations for the profile class.

764 **Table 28 – Operations: CIM_OwningCollectionElement**

Operation	Requirement	Messages
Associators	Unspecified	None
AssociatorNames	Unspecified	None
References	Unspecified	None
ReferenceNames	Unspecified	None

765 8.21 CIM_RegisteredProfile

766 All operations in the default list in 8.10 shall be implemented as defined in [DSP0200](#).

767 NOTE: Related profiles may define additional requirements on operations for the profile class.

768 8.22 CIM_ServiceAffectsElement

769 Table 29 lists implementation requirements for operations. If implemented, these operations shall be
 770 implemented as defined in [DSP0200](#). In addition, and unless otherwise stated in Table 29, all operations
 771 in the default list in 8.10 shall be implemented as defined in [DSP0200](#).

772 NOTE: Related profiles may define additional requirements on operations for the profile class.

773 **Table 29 – Operations: CIM_ServiceAffectsElement**

Operation	Requirement	Messages
Associators	Unspecified	None
AssociatorNames	Unspecified	None
References	Unspecified	None
ReferenceNames	Unspecified	None

774 8.23 CIM_UnsignedCredential

775 All operations in the default list in 8.10 shall be implemented as defined in [DSP0200](#).

776 NOTE: Related profiles may define additional requirements on operations for the profile class.

777 8.24 CIM_X509Certificate

778 Table 30 lists implementation requirements for operations. If implemented, these operations shall be
 779 implemented as defined in [DSP0200](#). In addition, and unless otherwise stated in Table 30, all operations
 780 in the default list in 8.10 shall be implemented as defined in [DSP0200](#).

781 NOTE: Related profiles may define additional requirements on operations for the profile class.

782

Table 30 – Operations: CIM_X509Certificate

Operation	Requirement	Messages
ModifyInstance	Optional	See 8.24.1.
DeleteInstance	Optional	See 8.24.2.

783 **8.24.1 CIM_X509Certificate — ModifyInstance**

784 If the SupportedMethods property of the CIM_CertificateManagementCapabilities instance associated
 785 with the Managing Service of the CIM_X509Certificate instance contains a value 107 (Modify
 786 X509Certificate), then the ModifyInstance operation shall be supported.

787 If the invalidation of the certificates is supported, the IsValid property shall be modified using
 788 theModifyInstance operation.

789 The ModifyInstance operation shall fail if the Activities property of the CIM_AssociatedPrivilege,
 790 referencing the CIM_Identity instance representing the executor’s security principal, does not contain a
 791 value of 6 (Write).

792 **8.24.2 CIM_X509Certificate — DeleteInstance**

793 If the SupportedMethods property of the CIM_CertificateManagementCapabilities instance associated
 794 with the Managing Service of the CIM_X509Certificate instance contains a value of 108 (Delete
 795 X509Certificate), then the DeleteInstance operation shall be supported.

796 If the removal of the certificates from the key store is supported, the DeleteInstance operation shall be
 797 supported on the CIM_X509Certificate class.

798 The ModifyInstance operation shall fail if the Activities property of the CIM_AssociatedPrivilege,
 799 referencing the CIM_Identity instance representing the executor’s security principal, does not contain a
 800 value of 3 (Delete).

801 **8.25 CIM_X509CRL**

802 All operations in the default list in 8.10 shall be implemented as defined in [DSP0200](#).

803 NOTE: Related profiles may define additional requirements on operations for the profile class.

804 **8.26 CIM_Identity**

805 All operations in the default list in 8.10 shall be implemented as defined in [DSP0200](#).

806 NOTE: Related profiles may define additional requirements on operations for the profile class.

807 **8.27 CIM_AssociatedPrivilege**

808 Table 31 lists implementation requirements for operations. If implemented, these operations shall be
 809 implemented as defined in [DSP0200](#). In addition, and unless otherwise stated in Table 31, all operations
 810 in the default list in 8.10 shall be implemented as defined in [DSP0200](#).

811 NOTE: Related profiles may define additional requirements on operations for the profile class.

812

Table 31 – Operations: CIM_AssociatedPrivilege

Operation	Requirement	Messages
Associators	Unspecified	None
AssociatorNames	Unspecified	None
References	Unspecified	None
ReferenceNames	Unspecified	None

813

814 8.28 CIM_ConcreteDependency

815 Table 31 lists implementation requirements for operations. If implemented, these operations shall be
 816 implemented as defined in [DSP0200](#). In addition, and unless otherwise stated in Table 31, all operations
 817 in the default list in 8.10 shall be implemented as defined in [DSP0200](#).

818 NOTE: Related profiles may define additional requirements on operations for the profile class.

819

Table 32 – Operations: CIM_ConcreteDependency

Operation	Requirement	Messages
Associators	Unspecified	None
AssociatorNames	Unspecified	None
References	Unspecified	None
ReferenceNames	Unspecified	None

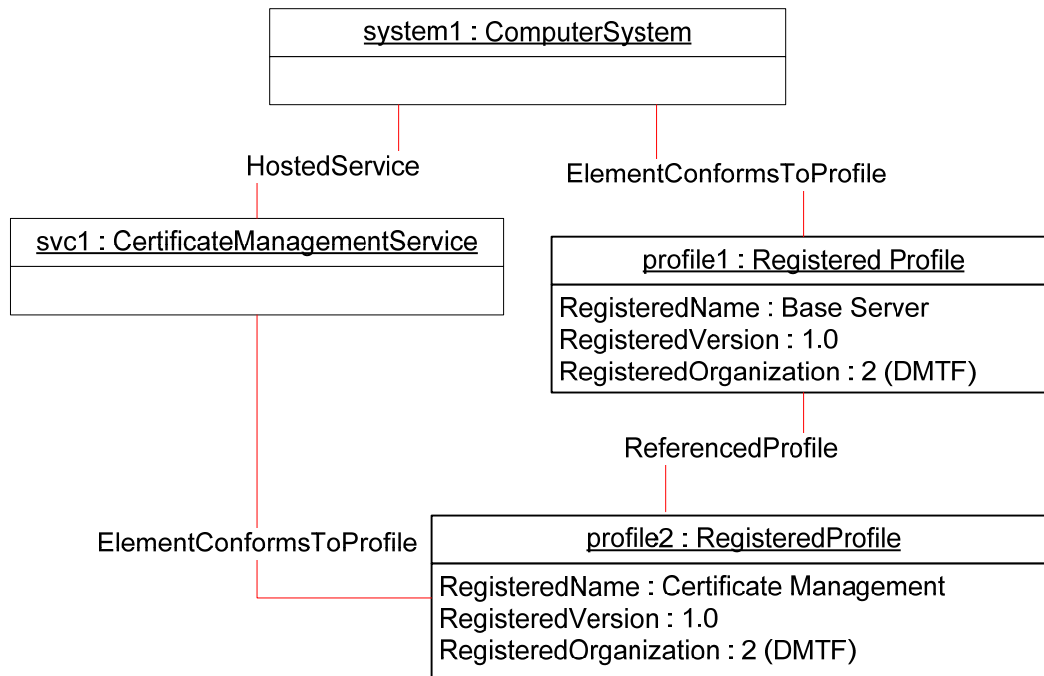
820

821 9 Use cases

822 This clause contains object diagrams and use cases for the *Certificate Management Profile*. The contents
 823 of this clause are for informative purposes only and do not constitute normative requirements for
 824 implementations of this specification.

825 9.1 Profile registration

826 Figure 2 describes one of the ways that the implementation can advertise the instantiation of the
 827 *Certificate Management Profile*. Using scoping instance methodology as described in the *Profile*
 828 *Registration Profile*, profile2 contains the version information for the *Certificate Management Profile*
 829 implementation.



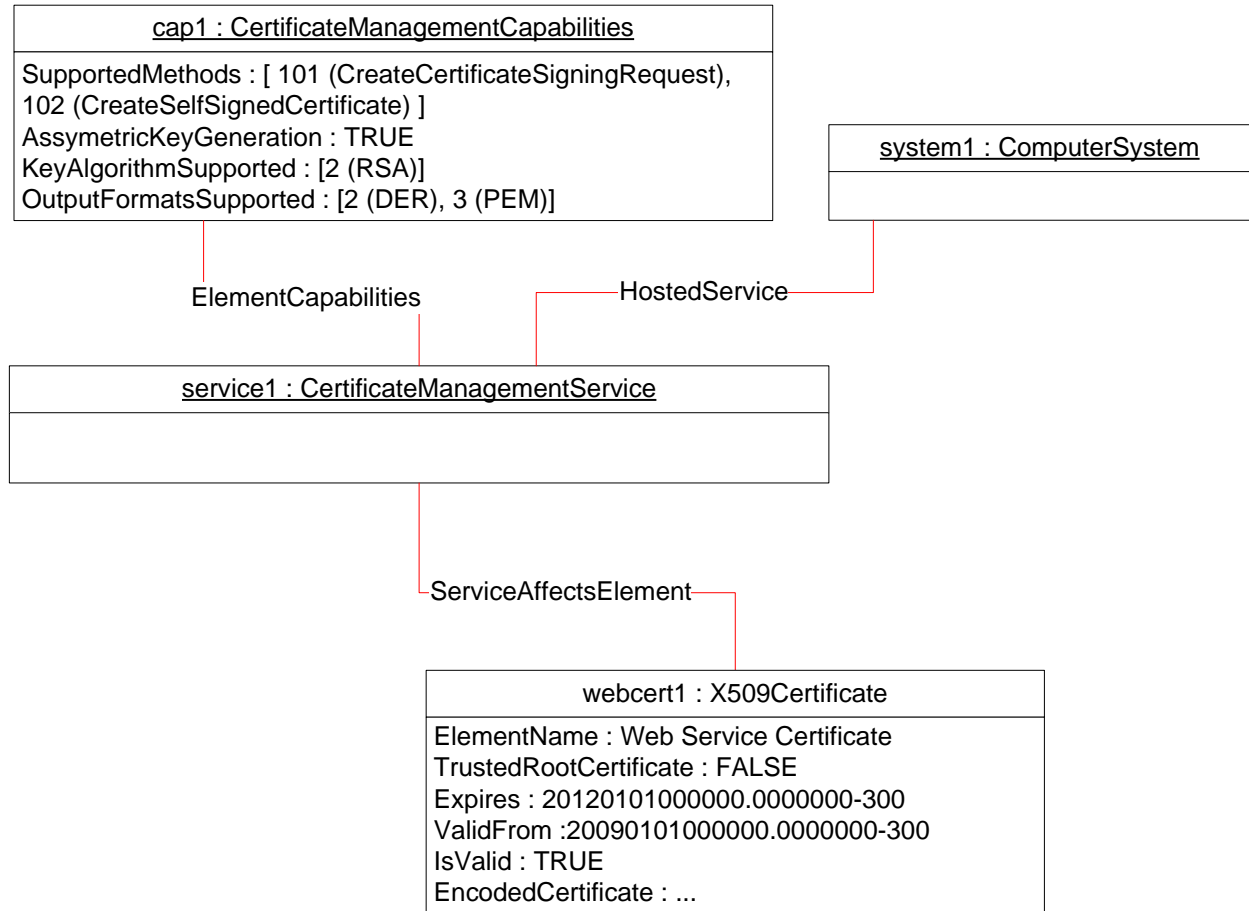
830

831

Figure 2 – Profile registration

832 **9.2 Simple certificate management**

833 Figure 3 represents a minimal instantiation of the *Certificate Management Profile* without key stores and
 834 certificate utilizers. Webcert1 is the only certificate managed by the service. The managed element that
 835 uses the webcert1 certificate is not instantiated, and thus the CredentialContext association does not
 836 reference the webcert1 instance. Instead, the webcert1.ElementName property describes the purpose of
 837 the certificate. Cap1 represents the capabilities of service1. Webcert1 is the only certificate that is being
 838 managed by service1. The managed element that uses the webcert1 certificate is not instantiated, and
 839 thus the CredentialContext association does not reference the webcert1 instance. Instead, the
 840 webcert1.ElementName property describes the purpose of the certificate.



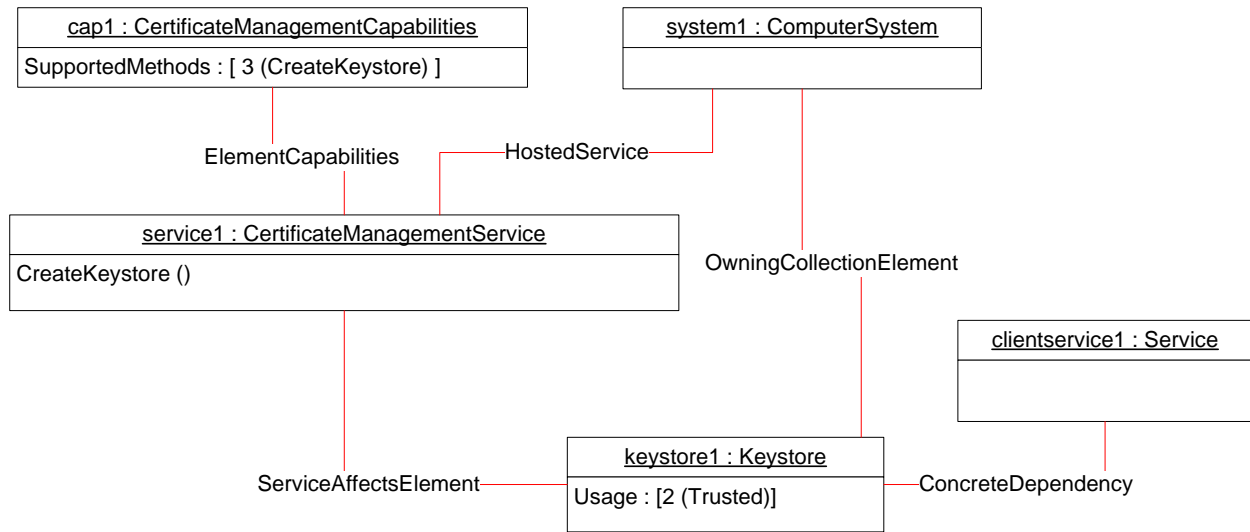
841

842

Figure 3 – Simple certificate management

843 **9.3 Keystore**

844 Figure 4 represents an instantiation of the *Certificate Management Profile*. In this instantiation, the
 845 instrumentation advertises the ability to create new key stores using the cap1.SupportedMethods
 846 property, which contains the value 3 (CreateKeystore). Upon successful invocation of a similarly named
 847 method on service1, a new key store will be created and will be represented by the keystore1 instance.
 848 Keystore1 properties will be equal to the Keystore embedded instance parameter's property of the
 849 invoked CreateKeystore method. The association instances referencing keystore1 are created based on
 850 the other parameters of the invoked CreateKeystore method. (See 8.2 and the method description in the
 851 CIM schema.)



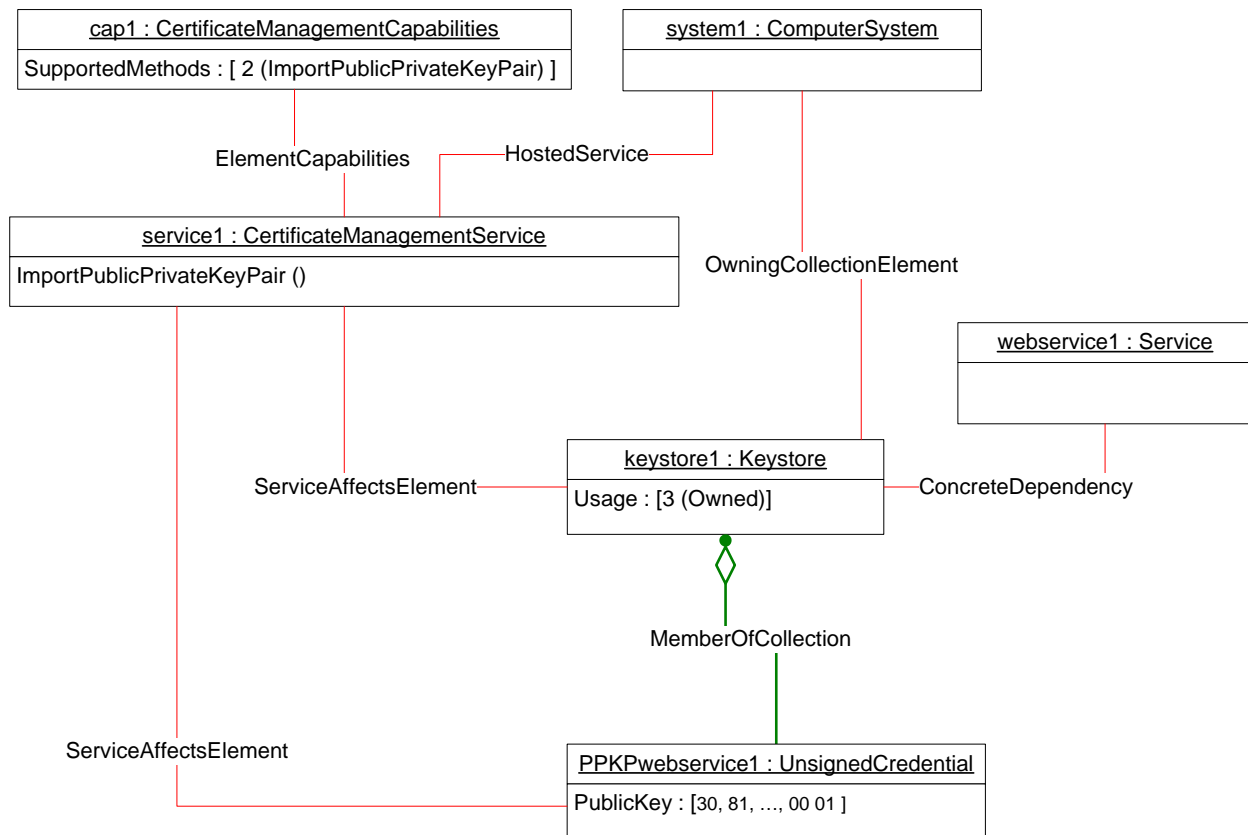
852

853

Figure 4 – Key store

854 **9.4 Import asymmetric key**

855 Figure 5 represents an instantiation of the *Certificate Management Profile*. In this instantiation, the
 856 instrumentation advertises the ability to import public/private key pairs using the cap1.SupportedMethods
 857 property, which contains the value 2 (ImportPublicPrivateKeyPair). Upon successful invocation of a
 858 similarly named method on service1, a new asymmetric key pair represented by PPKPwebservice1 will
 859 be inserted in the key store represented by keystore1. The PPKPwebservice1.PublicKey property will
 860 match the PublicKey parameter of the method. PPKPwebservice1 will not expose the private key for
 861 security purposes. The association instances referencing PPKPwebservice1 are created based on the
 862 other parameters of the invoked method. (See 8.1 and the method description in the CIM schema.)



863

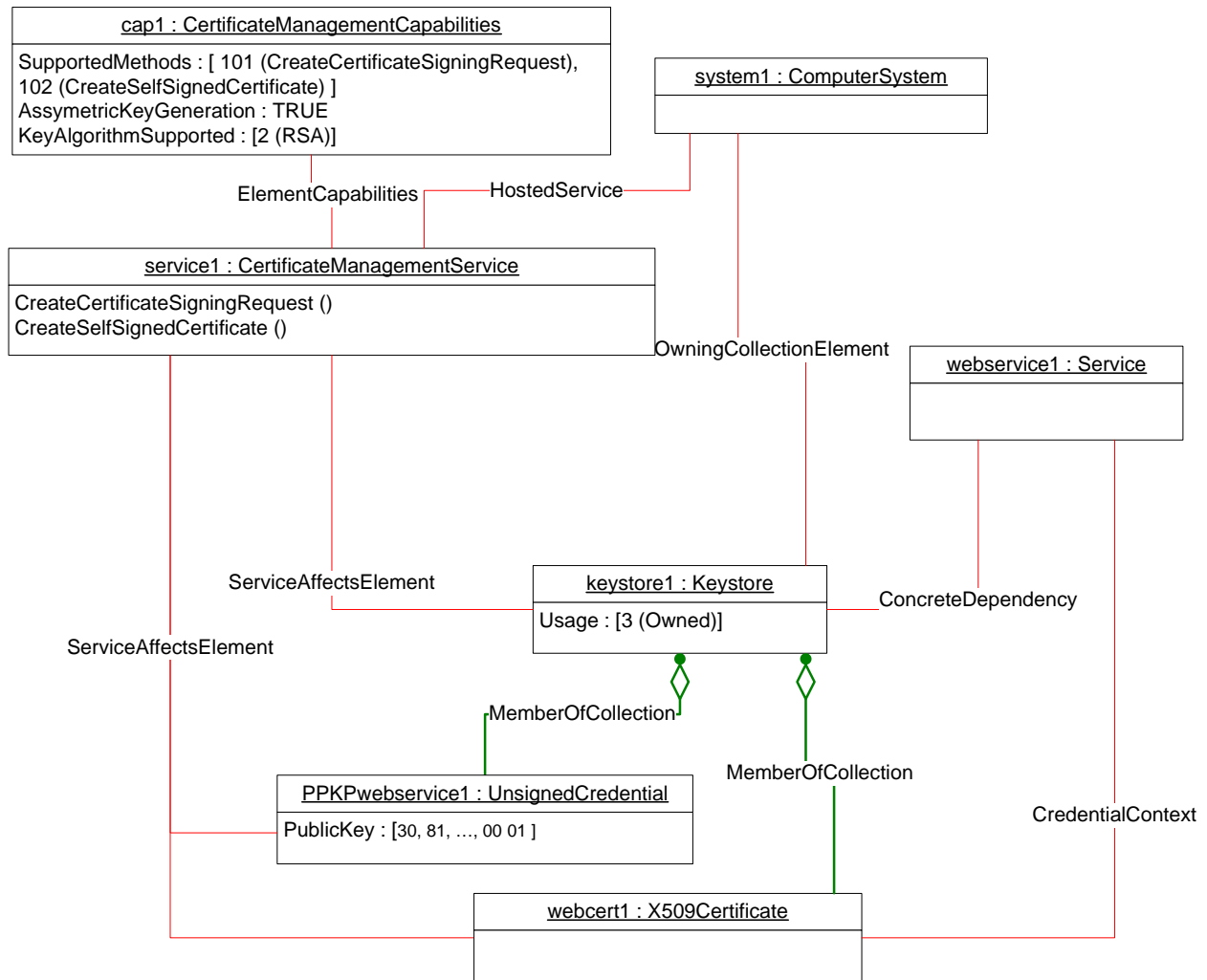
864

Figure 5 – Importing public/private key pair

865 9.5 CSR and self-signed certificate

866 Figure 6 represents an instantiation of the *Certificate Management Profile*. In this instantiation, the
 867 instrumentation advertises the ability to create CSRs and self-signed certificates using the
 868 cap1.SupportedMethods property, which contains the values 101 (CreateCertificateSigningRequest) and
 869 102 (CreateSelfSignedCertificate). The cap1.AsymmetricKeyGeneration property advertises the capability
 870 of service1 to generate asymmetric keys, specifically so that if the Subject, PublicKeyAlgorithm, has
 871 PublicKeySize parameters that are valid non-NULL values, the service will be able to create either CSR
 872 or the self-signed certificate generating a public/private asymmetric key pair. KeyAlgorithmSupported
 873 advertises the value maps that could be used as the value for the PublicKeyAlgorithm parameter. If
 874 AsymmetricKeyGeneration is FALSE, to create CSR or a self-signed certificate the PublicPrivateKeyPair
 875 parameter needs to reference an instance of CIM_UnsignedCredential, such as PPKPwebservice1, that
 876 represents a previously imported public/private asymmetric key pair.

877 Upon successful invocation of the CreateCertificateSigningRequest method on service1, the CSR will be
 878 returned as the CSR[] output parameter. Upon successful invocation of the CreateSelfSignedCertificate
 879 method, a new instance of CIM_X509Certificate, webcert1, will be instantiated, and the reference of the
 880 instance will be returned as the SelfSignedCertificate output parameter. The association instances
 881 referencing webcert1 are created based on the other parameters of the invoked method. (See 8.3, 8.4,
 882 and the method descriptions in the CIM schema.)



883

884

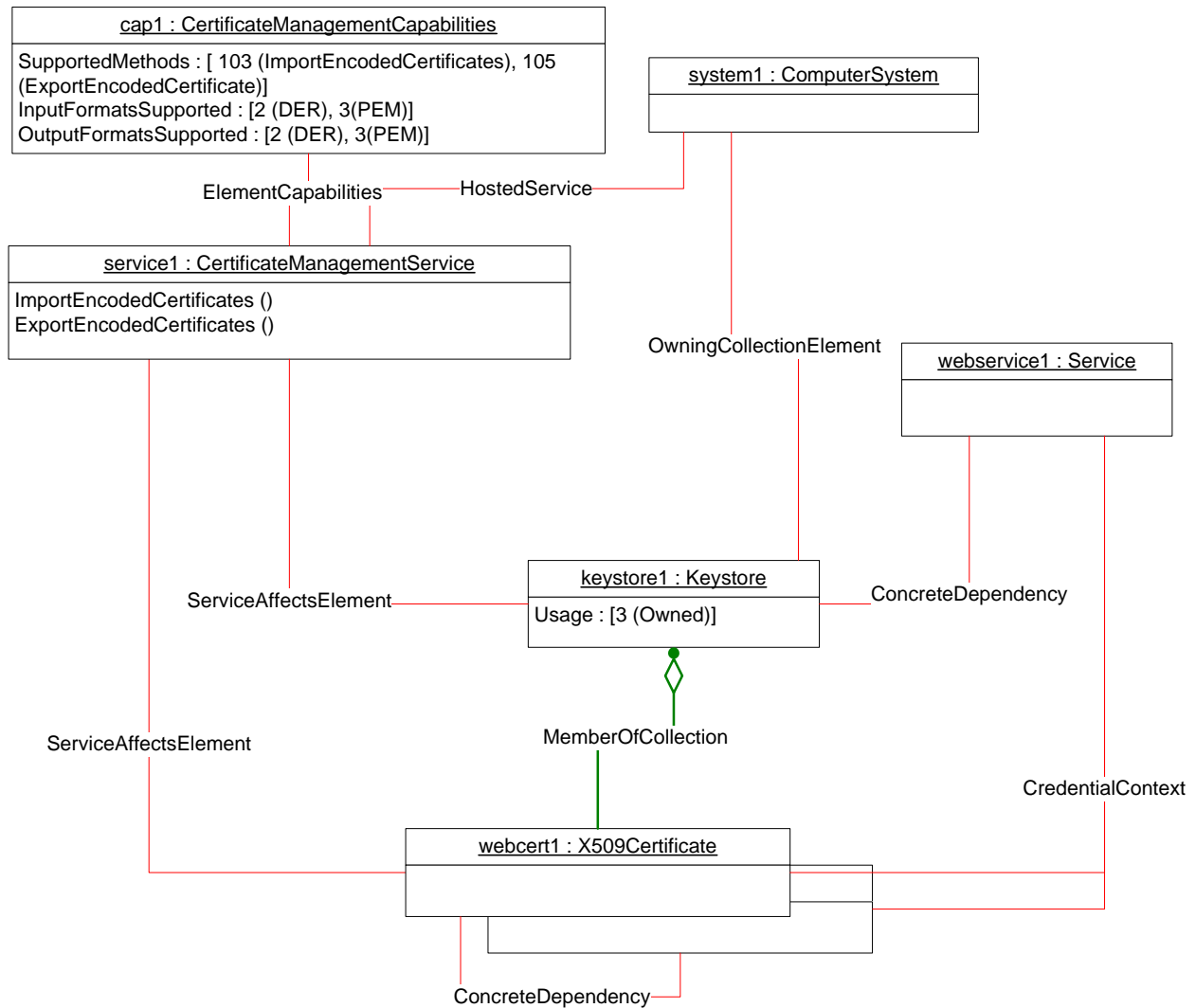
Figure 6 – Creating CSR and self-signed certificate

885 9.6 Import and export of certificates

886 Figure 7 represents an instantiation of the *Certificate Management Profile*. In this instantiation, the
 887 instrumentation advertises the ability to export and import certificates using the cap1.SupportedMethods
 888 property, which contains the values 103 (ImportEncodedCertificates) and 105 (ExportEncodedCertificate).
 889 The cap1.InputFormatsSupported property advertises the capability of service1 to import certificates in
 890 the specified formats and the value of the Format parameter on the InputEncodedCertificates method will
 891 match one of the values in this property array. Upon successful invocation of the
 892 ImportEncodedCertificates method a new instance of CIM_X509Certificate, webcert1, will be instantiated
 893 and the reference of the instance returned as the NewCertificates[] output parameter. The association
 894 instances referencing webcert1 are created based on the other parameters of the invoked method. (See
 895 8.5 and the method descriptions in the CIM schema.)

896 The cap1.OutputFormatsSupported property advertises the capability of service1 to export certificates in
 897 the specified formats, and the value of the Format parameter on the ExportEncodedCertificates() method
 898 will match one of the values in this property array. In this case, the CertificatestoExport parameter
 899 references webcert1 and upon successful invocation of the ExportEncodedCertificates() method the
 900 EncodedCertificates[] array will contain the formatted version of webcert1.

901



902

903

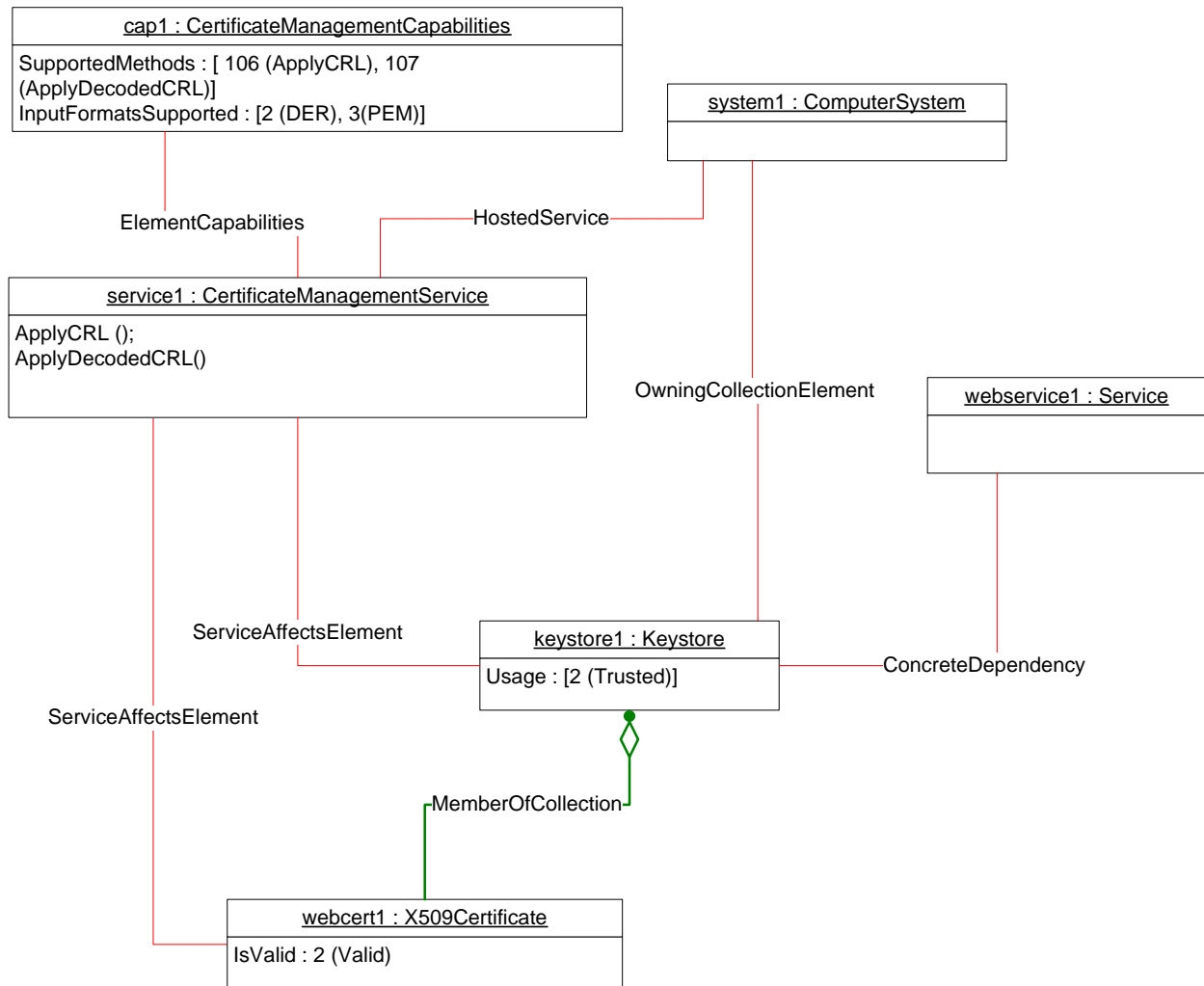
Figure 7 – Import and export of certificates

904 **9.7 CRL**

905 Figure 8 and Figure 9 represent an instantiation of the *Certificate Management Profile*.

906 **9.7.1 Before applying the CRL**

907 In this instantiation, the instrumentation advertises the ability to apply CRLs to key stores using the
 908 cap1.SupportedMethods property, which contains the value 106 (ApplyCRL). The
 909 cap1.InputFormatsSupported property advertises the capability of service1 to import CRLs in the
 910 specified formats, and the value of the Format parameter on the ApplyCRL method will match one of the
 911 values in this property array. The EncodedCRL[] parameter contains the encoded CRL, which consists of
 912 serial numbers of revoked certificates. In this case, one of these certificates is represented by webcert1.



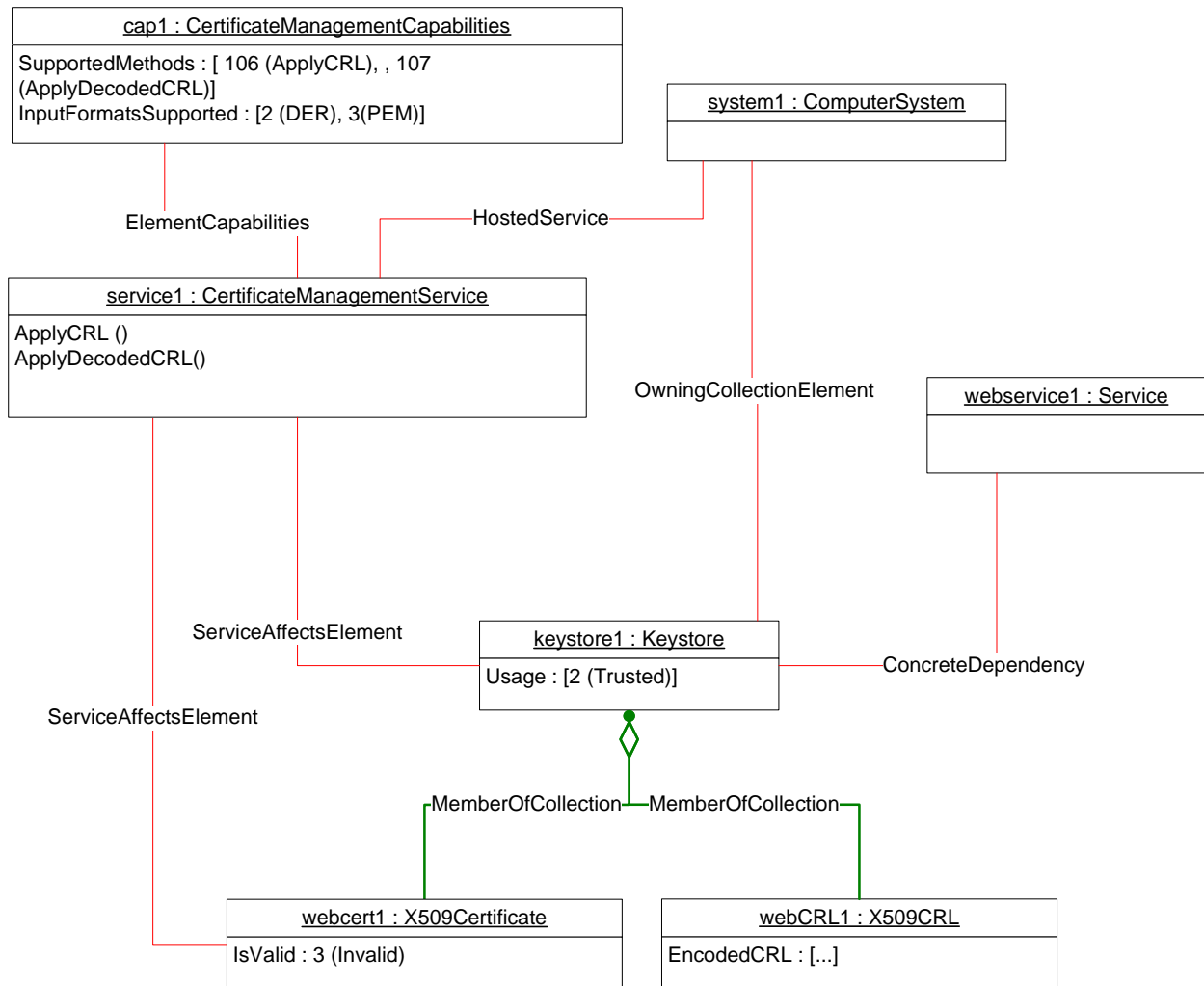
913

914

Figure 8 – Before CRL application

915 **9.7.2 After Applying the CRL**

916 Upon successful invocation of the ImportEncodedCertificates method a new instance of CIM_X509CRL,
 917 webCRL1, will be instantiated and the reference of the instance returned as the NewCertificates[] output
 918 parameter. The association instances referencing webCRL1 are created based on the other parameters
 919 of the invoked method. (See 8.8 and the method descriptions in the CIM schema.) Upon successful
 920 execution, the certificate represented by webcert1 will be revoked and the webcert1.IsValid property will
 921 be set to 3 (Invalid).



922

923

Figure 9 – After CRL application

924 **10 CIM elements**

925 Table 33 shows the instances of CIM elements for this profile. Instances of the CIM elements shall be
 926 implemented as described in Table 33. Clauses 7 (“Implementation”) and 8 (“Methods”) may impose
 927 additional requirements on these elements.

928

Table 33 – CIM elements: Certificate Management Profile

Element Name	Requirement	Description
Classes		
CIM_AssociatedPrivilege	Optional	See 10.18 and 7.4.
CIM_CertificateManagementCapabilities	Mandatory	See 10.1 and 7.2.2.
CIM_CertificateManagementService	Mandatory	See 10.2 and 7.2.
CIM_CredentialContext	Optional	See 10.3.
CIM_ConcreteDependency (CIM_Keystore)	Optional	See 10.4.

Element Name	Requirement	Description
CIM_ConcreteDependency (CIM_X509Certificate)	Optional	See 10.5.
CIM_ConcreteDependency	Optional	See 10.6
CIM_ElementCapabilities	Mandatory	See 10.7.
CIM_HostedService	Mandatory	See 10.8.
CIM_Keystore	Optional	See 7.1 and 10.9.
CIM_ServiceAffectsElement (CIM_Credential)	Optional	See 10.10.
CIM_MemberOfCollection	Conditional	See 10.11.
CIM_OwningCollectionElement	Conditional	See 10.12.
CIM_RegisteredProfile	Mandatory	See 10.13.
CIM_ServiceAffectsElement (CIM_Keystore)	Conditional	See 10.14.
CIM_UnsignedCredential	Mandatory	Zero or more instances shall be implemented. See 10.15.
CIM_X509Certificate	Mandatory	Zero or more instances shall be implemented. See 7.3 and 10.16.
CIM_X509CRL	Mandatory	Zero or more instances shall be implemented. See 10.17.
Indications		
None defined in this profile		

929 **10.1 CIM_CertificateManagementCapabilities**

930 CIM_CertificateManagementCapabilities is used to represent the capabilities of the
 931 CIM_CertificateManagementService. Table 34 details the requirements for instances of
 932 CIM_CertificateManagementCapabilities.

933 **Table 34 – Class: CIM_CertificateManagementCapabilities**

Elements	Requirement	Notes
InstanceID	Mandatory	Key
ElementName	Mandatory	Pattern ".*"
SupportedMethods	Mandatory	None
AsymmetricKeyGeneration	Mandatory	None
KeyAlgorithmSupported	Mandatory	None
InputFormatsSupported	Conditional	The property shall be supported if the SupportedMethods property has the values 103 (ImportEncodedCertificates) and 106 (ApplyCRL).

Elements	Requirement	Notes
OutputFormatsSupported	Conditional	The property shall be supported if the SupportedMethods property contains 105 (ExportEncodedCertificate).
SupportedSignatureAlgorithms	Optional	None
CumulativePrivilegeMethodology	Optional	None

934 10.2 CIM_CertificateManagementService

935 CIM_CertificateManagementService is used to manage PKI-based credentials represented by PKI
 936 Credential Instances. Table 35 details the requirements for instances of
 937 CIM_CertificateManagementService.

938 **Table 35 – Class: CIM_CertificateManagementService**

Elements	Requirement	Notes
SystemCreationClassName	Mandatory	Key
CreationClassName	Mandatory	Key
SystemName	Mandatory	Key
Name	Mandatory	Key
ElementName	Mandatory	Pattern ".*"
ImportPublicPrivateKeyPair()	Conditional	See 8.1.
CreateKeystore()	Conditional	See 8.2.
CreateCertificateSigningRequest()	Conditional	See 8.3.
CreateSelfSignedCertificate()	Conditional	See 8.4.
ImportEncodedCertificates()	Conditional	See 8.5 .
ImportCertificates()	Conditional	See 8.6.
ExportEncodedCertificates()	Conditional	See 8.7.
ApplyCRL()	Conditional	See 8.8.
ApplyDecodedCRL()	Conditional	See 8.9.

939 10.3 CIM_CredentialContext

940 CIM_CredentialContext is used to associate the PKI Credential Instance with the instances of subclasses
 941 of CIM_ManagedElement that represent the managed element that utilizes the credential. Table 36
 942 details the requirements for instances of CIM_CredentialContext.

943

Table 36 – Class: CIM_CredentialContext

Elements	Requirement	Notes
ElementInContext	Mandatory	Key: This property shall be a reference to the PKI Credential Instance. Cardinality * indicating zero or more references
ElementProvidingContext	Mandatory	Key: This property shall be a reference to the instance of the subclass of CIM_ManagedElement. Cardinality * indicating zero or more references

944 **10.4 CIM_ConcreteDependency (CIM_Keystore)**

945 CIM_ConcreteDependency is used to associate instances of a CIM_ManagedElement subclass with an
 946 instance of CIM_Keystore that the managed element utilizes. Table 37 details the requirements for
 947 instances of CIM_ConcreteDependency.

948

Table 37 – Class: CIM_ConcreteDependency

Elements	Requirement	Notes
Antecedent	Mandatory	Key: This property shall be a reference to the CIM_ManagedElement. Cardinality * indicating zero or more references
Dependent	Mandatory	Key: This property shall be a reference to the CIM_Keystore. Cardinality * indicating zero or more references

949 **10.5 CIM_ConcreteDependency (CIM_X509Certificate)**

950 CIM_ConcreteDependency is used to associate an instance of CIM_X509Certificate with another
 951 instance of CIM_X509Certificate to show the certificate chain hierarchy. Table 38 details the requirements
 952 for instances of CIM_ConcreteDependency.

953

Table 38 – Class: CIM_ConcreteDependency

Elements	Requirement	Notes
Antecedent	Mandatory	Key: This property shall be a reference to the CIM_X509Certificate instance representing the signing certificate. Cardinality 0..1 indicating zero or one reference
Dependent	Mandatory	Key: This property shall be a reference to the CIM_X509Certificate instance representing the signed certificate. Cardinality * indicating zero or more references

954 **10.6 CIM_ConcreteDependency**

955 CIM_ConcreteDependency is used to associate an instance of CIM_UnsignedCredential with an instance
 956 of CIM_X509Certificate that was generated based on the imported CIM_UnsignedCredential instance
 957 representing public-private key pair. Table 40 details the requirements for instances of
 958 CIM_ConcreteDependency.

959

Table 39 – CIM_ConcreteDependency

Properties	Requirement	Notes
Antecedent	Mandatory	Key: Shall reference the instance of CIM_UnsignedCredential Cardinality 0..1 indicating zero or one reference
Dependent	Mandatory	Key: Shall reference the instance of CIM_X509Certificate Cardinality * indicating zero or more references

960 10.7 CIM_ElementCapabilities

961 CIM_ElementCapabilities is used to associate an instance of CIM_CertificateManagementService with an
 962 instance of CIM_CertificateManagementCapabilities that describes the capabilities of the service. Table
 963 40 details the requirements for instances of CIM_ElementCapabilities.

964

Table 40 – CIM_ElementCapabilities

Properties	Requirement	Notes
ManagedElement	Mandatory	Key: Shall reference the instance of CIM_CertificateManagementService Cardinality 1..* indicating one or more references
Capabilities	Mandatory	Key: Shall reference the instance of CIM_CertificateManagementCapabilities Cardinality 1 indicating one and only one reference

965 10.8 CIM_HostedService

966 CIM_HostedService is used to associate CIM_CertificateManagementService with the Scoping Class.
 967 Table 41 details the requirements for instances of CIM_HostedService.

968

Table 41 – Class: CIM_HostedService

Elements	Requirement	Notes
Antecedent	Mandatory	Key: This property shall be a reference to the Scoping Instance. Cardinality 1 indicating one and only one reference
Dependent	Mandatory	Key: This property shall be a reference to the Central Instance. Cardinality 1..* indicating one or more references

969 10.9 CIM_Keystore

970 CIM_Keystore is used to represent the key store that accumulates PKI-based credentials represented by
 971 PKI Credential Instances. Table 42 details the requirements for instances of CIM_Keystore.

972

Table 42 – Class: CIM_Keystore

Elements	Requirement	Notes
InstanceID	Mandatory	Key
ElementName	Mandatory	Pattern ".*"

Elements	Requirement	Notes
Usage	Mandatory	None

973 **10.10 CIM_ServiceAffectsElement (CIM_Credential)**

974 CIM_ServiceAffectsElement is used to associate an instance of CIM_CertificateManagementService with
 975 an instance of CIM_X509Certificate, CIM_X509CRL, or CIM_UnsignedCredential. If the CIM_Keystore
 976 instance exists, CIM_ServiceAffectsElement is optional.

977 Table 43 contains the requirements for elements of this class.

978 **Table 43 – Class: CIM_ServiceAffectsElement (CIM_Credential)**

Elements	Requirement	Notes
AffectedElement	Mandatory	Key: This property shall reference the instance of CIM_X509Certificate, CIM_X509CRL, or CIM_UnsignedCredential. Cardinality * indicating zero or more references
AffectingElement	Mandatory	Key: This property shall reference the instance of CIM_CertificateManagementService. Cardinality 1 indicating one reference
ElementAffects	Mandatory	Matches 5 (Manages)

979 **10.11 CIM_MemberOfCollection**

980 CIM_MemberOfCollection is used to associate PKI Credential Instances with the instance of
 981 CIM_Keystore representing the key store that accumulates the PKI-based credentials. If the PKI
 982 Credential Instance exists, the CIM_MemberOfCollection instance is mandatory.

983 Table 44 provides information about the properties of CIM_MemberOfCollection.

984 **Table 44 – Class: CIM_MemberOfCollection**

Properties	Requirement	Notes
GroupComponent	Mandatory	Key: This property shall reference an instance of CIM_Keystore. Cardinality 1 indicating one and only one reference
PartComponent	Mandatory	Key: This property shall reference PKI Credential Instances. Cardinality * indicating zero or more references

985 **10.12 CIM_OwningCollectionElement**

986 CIM_OwningCollectionElement is used to associate a CIM_Keystore instance with its scoping
 987 CIM_System instance. If the CIM_Keystore instance exists, the CIM_OwningCollectionElement is
 988 mandatory.

989 Table 45 provides information about the properties of CIM_OwningCollectionElement.

990 **Table 45 – Class: CIM_OwningCollectionElement**

Properties	Requirement	Notes
OwningElement	Mandatory	Key: This property shall reference the Scoping Instance of this profile. Cardinality 1 indicating one and only one reference
OwnedElement	Mandatory	Key: This property shall be an instance of CIM_Keystore. Cardinality * indicating zero or more references

991 **10.13 CIM_RegisteredProfile**

992 The CIM_RegisteredProfile class is defined by the [Profile Registration Profile](#). The requirements denoted
 993 in Table 46 are in addition to those mandated by the [Profile Registration Profile](#).

994 **Table 46 – Class: CIM_RegisteredProfile**

Elements	Requirement	Notes
RegisteredName	Mandatory	This property shall have a value of "Certificate Management".
RegisteredVersion	Mandatory	This property shall have a value of "1.0.0".
RegisteredOrganization	Mandatory	This property shall have a value of 2 ("DMTF").

995 **10.14 CIM_ServiceAffectsElement (CIM_Keystore)**

996 CIM_ServiceAffectsElement is used to associate an instance of CIM_CertificateManagementService with
 997 an instance of CIM_Keystore that represents a key store that could be managed using the service. If the
 998 CIM_Keystore instance exists, CIM_ServiceAffectsElement is mandatory.

999 Table 47 contains the requirements for elements of this class.

1000

Table 47 – Class: CIM_ServiceAffectsElement (CIM_Keystore)

Elements	Requirement	Notes
AffectedElement	Mandatory	Key: This property shall reference the instance of CIM_Keystore. Cardinality * indicating zero or more references
AffectingElement	Mandatory	Key: This property shall reference the instance of CIM_CertificateManagementService. Cardinality 1 indicating one reference.
ElementAffects	Mandatory	Matches 5 (Manages)

1001 **10.15 CIM_UnsignedCredential**

1002 CIM_UnsignedCredential is used to represent the Asymmetric key (public/private key pair). Table 48
1003 details the requirements for instances of CIM_UnsignedCredential.

1004

Table 48 – Class: CIM_UnsignedCredential

Elements	Requirement	Notes
InstanceID	Mandatory	Key
ElementName	Mandatory	Pattern ".*"
PublicKey	Mandatory	

1005 **10.16 CIM_X509Certificate**

1006 CIM_X509Certificate is used to represent the X509 certificate. Table 49 details the requirements for
1007 instances of CIM_X509Certificate.

1008

Table 49 – Class: CIM_X509Certificate

Elements	Requirement	Notes
InstanceID	Mandatory	Key
ElementName	Mandatory	Pattern ".*"
TrustedRootCertificate	Optional	
Expires	Optional	
ValidFrom	Optional	
IsValid	Optional	
EncodedCertificate	Mandatory	

1009 **10.17 CIM_X509CRL**

1010 CIM_X509CRL is used to represent the X509 CRL. Table 50 details the requirements for instances of
 1011 CIM_X509CRL.

1012 **Table 50 – Class: CIM_X509CRL**

Elements	Requirement	Notes
InstanceID	Mandatory	Key
ElementName	Mandatory	Pattern ".*"
Issued	Optional	
NextUpdate	Optional	
Issuer	Optional	If the SupportedMethods property array of the associated instance of CIM_CertificateManagementCapabilities contains the value 107 (ApplyDecodedCRL), the Issuer property shall be supported.
SerialNumbers	Conditional	If the SupportedMethods property array of the associated instance of CIM_CertificateManagementCapabilities contains the value 107 (ApplyDecodedCRL), the Issuer property shall be supported.
EncodedCRL	Conditional	If the SupportedMethods property array of the associated instance of CIM_CertificateManagementCapabilities contains the value 106 (ApplyCRL), the Issuer property shall be supported.

1013 **10.18 CIM_AssociatedPrivilege**

1014 CIM_AssociatedPrivilege is used to associate the security principal with the credential or credential store
 1015 to which the security principal has access authorization.

1016 Table 51 lists requirements that are in addition to those enumerated in the [Credential Management](#)
 1017 [Profile](#).

1018 **Table 51 – Class: CIM_AssociatedPrivilege**

Elements	Requirement	Notes
Activities	Mandatory	See clause 7.4.

1019

1020

**ANNEX A
(informative)**

Change Log

1021
1022
1023
1024
1025

Version	Date	Description
1.0.0	2011-09-16	DMTF Standard

1026
1027