



1
2
3
4

Document Number: DSP0243

Date: 2013-08-22

Version: 2.0.1

5 **Open Virtualization Format Specification**

6 **Document Type: Specification**

7 **Document Status: DMTF Standard**

8 **Document Language: en-US**

9 Copyright notice

10 Copyright © 2009-2010, 2012-2013 Distributed Management Task Force, Inc. (DMTF). All rights
11 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26 withdrawn or modified after publication, and shall be indemnified and held harmless by any party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>.

CONTENTS

33	Foreword	6
34	Introduction.....	8
35	1 Scope	9
36	2 Normative References.....	9
37	3 Terms and Definitions	10
38	4 Symbols and Abbreviated Terms	12
39	5 OVF Packages	12
40	5.1 OVF Package Structure	12
41	5.2 Virtual Disk Formats.....	14
42	5.3 Distribution as a Single File	14
43	5.4 Distribution as a Set of Files	15
44	6 OVF Descriptor.....	15
45	7 Envelope Element	16
46	7.1 File References	17
47	7.2 Content Element	18
48	7.3 Extensibility	19
49	7.4 Conformance	20
50	8 Virtual Hardware Description.....	21
51	8.1 VirtualHardwareSection	21
52	8.2 Extensibility	23
53	8.3 Virtual Hardware Elements	23
54	8.4 Ranges on Elements.....	26
55	9 Core Metadata Sections in version 2	28
56	9.1 DiskSection	29
57	9.2 NetworkSection.....	30
58	9.3 ResourceAllocationSection	31
59	9.4 AnnotationSection.....	32
60	9.5 ProductSection.....	32
61	9.6 EulaSection.....	35
62	9.7 StartupSection	36
63	9.8 DeploymentOptionSection	37
64	9.9 OperatingSystemSection	39
65	9.10 InstallSection.....	39
66	9.11 EnvironmentFilesSection	40
67	9.12 BootDeviceSection.....	41
68	9.13 SharedDiskSection	42
69	9.14 ScaleOutSection	43
70	9.15 PlacementGroupSection and PlacementSection.....	44
71	9.16 Encryption Section	46
72	10 Internationalization	48
73	10.1 Internal Resource Bundles	48
74	10.2 External Resource Bundles	49
75	10.3 Message Content in External File.....	49
76	11 OVF Environment.....	50
77	11.1 Environment Document	50
78	11.2 Transport.....	51
79	ANNEX A (informative) Symbols and Conventions	53
80	ANNEX B (normative) OVF XSD	54
81	ANNEX C (informative) OVF Mime Type Registration Template	55
82	ANNEX D (informative) Network Port Profile Examples	57

83 D.1 Example 1 (OVF Descriptor for One Virtual System and One Network with an Inlined
84 Network Port Profile)..... 57
85 D.2 Example 2 (OVF Descriptor for One Virtual System and One Network with a Locally
86 Referenced Network Port Profile) 60
87 D.3 Example 3 (OVF Descriptor for One Virtual System and One Network with a Network
88 Port Profile referenced by a URI)..... 63
89 D.4 Example 4 (OVF Descriptor for Two Virtual Systems and One Network with Two
90 Network Port Profiles referenced by URIs) 66
91 D.5 Example 5 (networkportprofile1.xml) 71
92 Example 6 (networkportprofile2.xml) 71
93 D.6 71
94 ANNEX E (informative) Change Log..... 73
95 Bibliography 74
96

97 **Tables**

98 Table 1 – XML Namespace Prefixes 16

99 Table 2 – Actions for Child Elements with `ovf:required` Attribute..... 23

100 Table 3 – HostResource Element 25

101 Table 4 – Elements for Virtual Devices and Controllers 25

102 Table 5 – Core Metadata Sections in version 2 28

103 Table 6 – Property Types 34

104 Table 7 – Property Qualifiers 35

105 Table 8 – Core Sections..... 51

106

107

Foreword

108 The *Open Virtualization Format Specification* (DSP0243) was prepared by the OVF Work Group of the
109 DMTF.

110 This specification has been developed as a result of joint work with many individuals and teams,
111 including:

112		
113	Lawrence Lamers	VMware Inc. (Chair)
114	Hemal Shah	Broadcom Corporation (co-Editor)
115	Steffen Grarup	VMware Inc. (co-Editor)
116		
117	Vincent Kowalski	BMC Software
118	Hemal Shah	Broadcom Corporation
119	John Crandall	Brocade Communications Systems
120	Marvin Waschke	CA Technologies
121	Naveen Joy	Cisco
122	Steven Neely	Cisco
123	Shishir Pardikar	Citrix Systems Inc.
124	Thomas Root	Citrix Systems Inc.
125	Richard Landau	DMTF Fellow
126	Jacques Durand	Fujitsu
127	Derek Coleman	Hewlett-Packard Company
128	Robert Freund	Hitachi, Ltd.
129	Fred Maciel	Hitachi, Ltd.
130	Eric Wells	Hitachi, Ltd.
131	Abdellatif Touimi	Huawei
132	Jeff Wheeler	Huawei
133	HengLiang Zhang	Huawei
134	Oliver Benke	IBM
135	Ron Doyle	IBM
136	Michael Gering	IBM
137	Michael Johanssen	IBM
138	Andreas Maier	IBM
139	Marc-Arthur Pierre-Louis	IBM
140	John Leung	Intel Corporation
141	Nitin Bhat	Microsoft Corporation
142	Maurizio Carta	Microsoft Corporation
143	Monica Martin	Microsoft Corporation
144	John Parchem	Microsoft Corporation
145	Ed Reed	Microsoft Corporation
146	Nihar Shah	Microsoft Corporation
147	Cheng Wei	Microsoft Corporation
148	Narayan Venkat	NetApp
149	Tatyana Bagerman	Oracle
150	Srinivas Maturi	Oracle
151	Dr. Fermín Galán Márquez	Telefónica
152	Miguel Ángel Peñalvo	Telefónica
153	Dr. Fernando de la Iglesia	Telefónica
154	Álvaro Polo	Telefónica
155	Steffen Grarup	VMware Inc.
156	Lawrence Lamers	VMware Inc.
157	Rene Schmidt	VMware Inc.
158	Paul Ferdinand	WBEM Solutions

159	Junsheng Chu	ZTE Corporation
160	Bhumip Khasnabish	ZTE Corporation
161	Ghazanfar Ali	ZTE Corporation

162

Introduction

163 The *Open Virtualization Format (OVF) Specification* describes an open, secure, portable, efficient and
164 extensible format for the packaging and distribution of software to be run in virtual machines. The key
165 properties of the format are as follows:

- 166 • **Optimized for distribution**

167 OVF supports content verification and integrity checking based on industry-standard public key
168 infrastructure, and it provides a basic scheme for management of software licensing.

- 169 • **Optimized for a simple, automated user experience**

170 OVF supports validation of the entire package and each virtual machine or metadata
171 component of the OVF during the installation phases of the virtual machine (VM) lifecycle
172 management process. It also packages with the package relevant user-readable descriptive
173 information that a virtualization platform can use to streamline the installation experience.

- 174 • **Supports both single VM and multiple-VM configurations**

175 OVF supports both standard single VM packages and packages containing complex, multi-tier
176 services consisting of multiple interdependent VMs.

- 177 • **Portable VM packaging**

178 OVF is virtualization platform neutral, while also enabling platform-specific enhancements to be
179 captured. It supports the full range of virtual hard disk formats used for hypervisors today, and it
180 is extensible, which allow it to accommodate formats that may arise in the future. Virtual
181 machine properties are captured concisely and accurately.

- 182 • **Vendor and platform independent**

183 OVF does not rely on the use of a specific host platform, virtualization platform, or guest
184 operating system.

- 185 • **Extensible**

186 OVF is immediately useful — and extensible. It is designed to be extended as the industry
187 moves forward with virtual appliance technology. It also supports and permits the encoding of
188 vendor-specific metadata to support specific vertical markets.

- 189 • **Localizable**

190 OVF supports user-visible descriptions in multiple locales, and it supports localization of the
191 interactive processes during installation of an appliance. This capability allows a single
192 packaged appliance to serve multiple market opportunities.

- 193 • **Open standard**

194 OVF has arisen from the collaboration of key vendors in the industry, and it is developed in an
195 accepted industry forum as a future standard for portable virtual machines.

196 It is not an explicit goal for OVF to be an efficient execution format. A hypervisor is allowed but not
197 required to run software in virtual machines directly out of the Open Virtualization Format.

198

Open Virtualization Format Specification

199 1 Scope

200 The *Open Virtualization Format (OVF) Specification* describes an open, secure, portable, efficient and
201 extensible format for the packaging and distribution of software to be run in virtual machines.

202 This version of the specification (2.0) is intended to allow OVF 1.x tools to work with OVF 2.0 descriptors
203 in the following sense:

- 204 • Existing OVF 1.x tools should be able to parse OVF 2.0 descriptors.
- 205 • Existing OVF 1.x tools should be able to give warnings/errors if dependencies to 2.0 features are
206 required for correct operation.

207 2 Normative References

208 The following referenced documents are indispensable for the application of this document. For dated
209 references, only the edition cited applies. For undated references, the latest edition of the referenced
210 document (including any amendments) applies.

211 [ISO/IEC/IEEE 9945:2009](http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=50516): Information technology -- Portable Operating System Interface (POSIX®) Base
212 Specifications, Issue 7

213 http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=50516

214 DMTF DSP0004, *Common Information Model (CIM) Infrastructure Specification 2.7*,

215 http://www.dmtf.org/standards/published_documents/DSP0004_2.7.pdf

216 DMTF DSP0230, *WS-CIM Mapping Specification 1.1*,

217 http://www.dmtf.org/standards/published_documents/DSP0230_1.1.pdf

218 DMTF DSP1041, *Resource Allocation Profile (RAP) 1.1*,

219 http://www.dmtf.org/standards/published_documents/DSP1041_1.1.pdf

220 DMTF DSP1043, *Allocation Capabilities Profile (ACP) 1.0*,

221 http://www.dmtf.org/standards/published_documents/DSP1043_1.0.pdf

222 DMTF DSP1047, *Storage Resource Virtualization Profile 1.0*,

223 http://www.dmtf.org/standards/published_documents/DSP1047_1.0.pdf

224 DMTF DSP8023, *Open Virtualization Format (OVF) 2 XML Schema*,

225 <http://schemas.dmtf.org/ovf/envelope/2/dsp8023.xsd>

226 DMTF DSP8049, *Network Port Profile XML Schema*,

227 <http://schemas.dmtf.org/ovf/networkportprofile/1/dsp8049.xsd>

228 IETF RFC1738, T. Berners-Lee, *Uniform Resource Locators (URL)*, December 1994,

229 <http://tools.ietf.org/html/rfc1738>

230 IETF RFC1952, P. Deutsch, *GZIP file format specification version 4.3*, May 1996,

231 <http://tools.ietf.org/html/rfc1952>

232 IETF Standard 68, *Augmented BNF for Syntax Specifications: ABNF*,

233 <http://tools.ietf.org/html/rfc5234>

- 234 IETF RFC2616, R. Fielding et al, *Hypertext Transfer Protocol – HTTP/1.1*, June 1999,
235 <http://tools.ietf.org/html/rfc2616>
- 236 IETF Standard 66, *Uniform Resource Identifiers (URI): Generic Syntax*,
237 <http://tools.ietf.org/html/rfc3986>
- 238 ISO 9660, 1988 Information processing-Volume and file structure of CD-ROM for information interchange,
239 http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=17505
- 240 ISO, ISO/IEC Directives, Part 2, *Rules for the structure and drafting of International Standards*,
241 <http://isotc.iso.org/livelink/livelink.exe?func=ll&objId=4230456&objAction=browse&sort=subtype>
- 242 W3C, *XML Schema Part 1: Structures Second Edition*, 28 October 2004. W3C Recommendation. URL:
243 <http://www.w3.org/TR/2004/REC-xmlschema-1-20041028/>
- 244 W3C, *XML Schema Part 2: Datatypes Second Edition*, 28 October 2004. W3C Recommendation. URL:
245 <http://www.w3.org/TR/2004/REC-xmlschema-2-20041028/>
- 246 XML Encryption Syntax and Processing Version 1.1, 13 March 2012, W3C Candidate Recommendation
247 <http://www.w3.org/TR/2012/CR-xmlenc-core1-20120313/>
- 248 FIPS 180-2: Secure Hash Standard (SHS)
249 <http://csrc.nist.gov/publications/fips/fips180-2/fips180-2withchangenotice.pdf>

250 3 Terms and Definitions

251 For the purposes of this document, the following terms and definitions apply.

252 3.1

253 **can**

254 used for statements of possibility and capability, whether material, physical, or causal

255 3.2

256 **cannot**

257 used for statements of possibility and capability, whether material, physical, or causal

258 3.3

259 **conditional**

260 indicates requirements to be followed strictly to conform to the document when the specified conditions
261 are met

262 3.4

263 **mandatory**

264 indicates requirements to be followed strictly to conform to the document and from which no deviation is
265 permitted

266 3.5

267 **may**

268 indicates a course of action permissible within the limits of the document

269 3.6

270 **need not**

271 indicates a course of action permissible within the limits of the document

- 272 **3.7**
273 **optional**
274 indicates a course of action permissible within the limits of the document
- 275 **3.8**
276 **shall**
277 indicates requirements to be followed strictly to conform to the document and from which no deviation is
278 permitted
- 279 **3.9**
280 **shall not**
281 indicates requirements to be followed strictly to conform to the document and from which no deviation is
282 permitted
- 283 **3.10**
284 **should**
285 indicates that among several possibilities, one is recommended as particularly suitable, without
286 mentioning or excluding others, or that a certain course of action is preferred but not necessarily required
- 287 **3.11**
288 **should not**
289 indicates that a certain possibility or course of action is deprecated but not prohibited
- 290 **3.12**
291 **appliance**
292 see [virtual appliance](#)
- 293 **3.13**
294 **deployment platform**
295 the product that installs an OVF package
- 296 **3.14**
297 **guest software**
298 the software that runs inside a virtual machine
299 The guest is typically an operating system and some user-level applications and services.
- 300 **3.15**
301 **OVF package**
302 OVF XML descriptor file accompanied by zero or more files
- 303 **3.16**
304 **OVF descriptor**
305 OVF XML descriptor file
- 306 **3.17**
307 **platform**
308 see [deployment platform](#)
- 309 **3.18**
310 **virtual appliance**
311 a service delivered as a complete software stack installed on one or more virtual machines
312 A virtual appliance is typically expected to be delivered in an OVF package.

313 **3.19**
314 **virtual hardware**
315 the processor, memory and I/O resources of a virtual computer system

316 **3.20**
317 **virtual machine**
318 as defined in System Virtualization Profile

319 **3.21**
320 **virtual machine collection**
321 a collection comprised of a set of virtual machines. This service component can be a:
322 - simple set of one or more virtual machines, or
323 - a complex service component built out of a combination of virtual machines and other virtual
324 machine collections that enables nested complex service components.

325 **4 Symbols and Abbreviated Terms**

326 The following symbols and abbreviations are used in this document.

327 **4.1**
328 **CIM**
329 Common Information Model

330 **4.2**
331 **IP**
332 Internet Protocol

333 **4.3**
334 **OVF**
335 Open Virtualization Format

336 **4.4**
337 **VM**
338 Virtual Machine

339 **5 OVF Packages**

340 **5.1 OVF Package Structure**

341 An OVF package shall consist of the following files:

- 342 • one OVF descriptor with extension `.ovf`
- 343 • zero or one OVF manifest with extension `.mf`
- 344 • zero or one OVF certificate with extension `.cert`
- 345 • zero or more disk image files
- 346 • zero or more additional resource files, such as ISO images

347 The file extensions `.ovf`, `.mf` and `.cert` shall be used.

348 EXAMPLE 1: The following list of files is an example of an OVF package:

```
349 package.ovf
350 package.mf
351 de-DE-resources.xml
352 vmdisk1.vmdk
353 vmdisk2.vhd
354 resource.iso
```

355 An OVF package can be stored as either a single unit or a set of files, as described in 5.3 and 5.4. Both
356 modes shall be supported.

357 An OVF package may have a manifest file containing the SHA digests of individual files in the package.
358 OVF packages authored according to this version of the specification shall use SHA256 digests; older
359 OVF packages are allowed to use SHA1. The manifest file shall have an extension `.mf` and the same
360 base name as the `.ovf` file and be a sibling of the `.ovf` file. If the manifest file is present, a consumer of
361 the OVF package shall verify the digests by computing the actual SHA digests and comparing them with
362 the digests listed in the manifest file. The manifest file shall contain SHA digests for all distinct files
363 referenced in the `References` element of the OVF descriptor, see clause 7.1, and for no other files.

364 The syntax definitions below use ABNF with the exceptions listed in ANNEX A.

365 The format of the manifest file is as follows:

```
366 manifest_file = *( file_digest )
367 file_digest  = algorithm "(" file_name ")" "=" sp digest nl
368 algorithm    = "SHA1" | "SHA256"
369 digest       = *( hex-digit )
370 hex-digit    = "0" | "1" | "2" | "3" | "4" | "5" | "6" | "7" | "8" | "9" | "a" |
371 "b" | "c" | "d" | "e" | "f"
372 sp           = %x20
373 nl           = %x0A
```

374 EXAMPLE 2: The following example show the partial contents of a manifest file:

```
375 SHA256(package.ovf)= 9902cc5ec4f4a00cabbff7b60d039263587ab430d5fdbc5cd5e8707391c90a4
376 SHA256(vmdisk.vmdk)= aab66c4d70e17cec2236a651a3fc618cafc5ec6424122904dc0b9c286fce40c2
```

377 An OVF package may be signed by signing the manifest file. The digest of the manifest file is stored in a
378 certificate file with extension `.cert` file along with the base64-encoded X.509 certificate. The `.cert` file
379 shall have the same base name as the `.ovf` file and be a sibling of the `.ovf` file. A consumer of the OVF
380 package shall verify the signature and should validate the certificate. The format of the certificate file shall
381 be as follows:

```
382 certificate_file = manifest_digest certificate_part
383 manifest_digest = algorithm "(" file_name ")" "=" sp signed_digest nl
384 algorithm       = "SHA1" | "SHA256"
385 signed_digest   = *( hex-digit)
386 certificate_part = certificate_header certificate_body certificate_footer
387 certificate_header = "-----BEGIN CERTIFICATE-----" nl
388 certificate_footer = "-----END CERTIFICATE-----" nl
389 certificate_body  = base64-encoded-certificate nl
390                   ; base64-encoded-certificate is a base64-encoded X.509
391                   ; certificate, which may be split across multiple lines
392 hex-digit        = "0" | "1" | "2" | "3" | "4" | "5" | "6" | "7" | "8" | "9" | "a"
393 | "b" | "c" | "d" | "e" | "f"
394 sp               = %x20
395 nl               = %x0A
```

396 EXAMPLE 3: The following list of files is an example of a signed OVF package:

```
397 package.ovf
398 package.mf
399 package.cert
400 de-DE-resources.xml
401 vmdisk1.vmdk
402 vmdisk2.vmdk
403 resource.iso
```

404 EXAMPLE 4: The following example shows the contents of a sample OVF certification file, where the SHA1 digest
405 of the manifest file has been signed with a 512 bit key:

```
406 SHA1(package.mf) = 7f4b8efb8fe20c06df1db68281a63f1b088e19dbf00e5af9db5e8e3e319de
407 7019db88a3bc699bab6ccd9e09171e21e88ee20b5255cec3fc28350613b2c529089
408 -----BEGIN CERTIFICATE-----
409 MIIBGjCCASwCAQQwDQYJKoZIhvcNAQEEBQAwoDELMakGA1UEBhMCQVUxDDAKBgNV
410 BAgtA1FMRDEbMBkGA1UEAxMSU1NMZWV5L3JzYSB0ZXN0IENBMB4XDtk1MTAwOTIz
411 MzIwNVVoXDTk4MDcwNTIzMzIwNVowYDELMAkGA1UEBhMCQVUxDDAKBgNVBAgtA1FM
412 RDEZMBcGA1UEChMQTWluY29tIFB0eS4gTHRkLjELMAkGA1UECxmCQ1MxGzAZBgNV
413 BAMTElNTTGVheSBkZW1vIHNLcnZlcjBcMA0GCsGSIb3DQEBAQUAA0sAMEgCQQC3
414 LCXcScWua0PFLkHBLm2VejqpA1F4RQ8q0VjRiPafjx/Z/aWH3ipdMVvuJGa/wFXb
415 /nDFLDlFwP+oCPwhBtVPAGMBAAEwDQYJKoZIhvcNAQEEBQADQQARNFsihWIjBzb0
416 DcsU0BvL2bvSwJrPEqFlkDq3F4M6EgutL9axEcANWgbbEdAvNJD1dmEmoWny27Pn
417 Ims6ZOZB
418 -----END CERTIFICATE-----
```

419 The manifest and certificate files, when present, shall not be included in the *References* section of the
420 OVF descriptor (see 7.1). This ensures that the OVF descriptor content does not depend on whether the
421 OVF package has a manifest or is signed, and the decision to add a manifest or certificate to a package
422 can be deferred to a later stage.

423 The file extensions `.mf` and `.cert` may be used for other files in an OVF package, as long as they do
424 not occupy the sibling URLs or path names where they would be interpreted as the package manifest or
425 certificate.

426 5.2 Virtual Disk Formats

427 OVF does not require any specific disk format to be used, but to comply with this specification the disk
428 format shall be given by a URI which identifies an unencumbered specification on how to interpret the
429 disk format. The specification need not be machine readable, but it shall be static and unique so that the
430 URI may be used as a key by software reading an OVF package to uniquely determine the format of the
431 disk. The specification shall provide sufficient information so that a skilled person can properly interpret
432 the disk format for both reading and writing of disk data. The URI should be resolvable.

433 5.3 Distribution as a Single File

434 An OVF package may be stored as a single file using the TAR format. The extension of that file shall be
435 `.ova` (open virtual appliance or application).

436 EXAMPLE: The following example shows a sample filename for an OVF package of this type:

```
437 D:\virtualappliances\myapp.ova
```

438 For OVF packages stored as single file, all file references in the OVF descriptor shall be relative-path
439 references and shall point to files included in the TAR archive. Relative directories inside the archive are
440 allowed, but relative-path references shall not contain “..” dot-segments.

441 Ordinarily, a TAR extraction tool would have to scan the whole archive, even if the file requested is found
442 at the beginning, because replacement files can be appended without modifying the rest of the archive.
443 Entries in an OVF TAR file shall exist only once.

444 In addition, the entries shall be in one of the following orders inside the archive:

- 445 1) OVF descriptor
446 2) The remaining files shall be in the same order as listed in the References section (see 7.1). Note
447 that any external string resource bundle files for internationalization shall be first in the
448 References section (see clause 10).

449

- 450 1) OVF descriptor
451 2) OVF manifest
452 3) OVF certificate
453 4) The remaining files shall be in the same order as listed in the References section (see 7.1).
454 Note that any external string resource bundle files for internationalization shall be first in the
455 References section (see clause 10).

456

- 457 1) OVF Descriptor
458 2) The remaining files shall be in the same order as listed in the References section (see 7.1).
459 Note that any external string resource bundle files for internationalization shall be first in the
460 References section (see clause 10).
461 3) OVF manifest
462 4) OVF certificate

463 For deployment, the ordering restriction ensures that it is possible to extract the OVF descriptor from an
464 OVF TAR file without scanning the entire archive. For generation, the ordering restriction ensures that an
465 OVF TAR file can easily be generated on-the-fly. The restrictions do not prevent OVF TAR files from
466 being created using standard TAR packaging tools.

467 The TAR format used shall comply with the USTAR (Uniform Standard Tape Archive) format as defined
468 by the [ISO/IEC/IEEE 9945:2009](#).

469 5.4 Distribution as a Set of Files

470 An OVF package can be made available as a set of files, for example on a standard Web server.

471 EXAMPLE: An example of an OVF package as a set of files on Web server follows:

```
472 http://mywebsite/virtualappliances/package.ovf  
473 http://mywebsite/virtualappliances/vmdisk1.vmdk  
474 http://mywebsite/virtualappliances/vmdisk2.vmdk  
475 http://mywebsite/virtualappliances/resource.iso  
476 http://mywebsite/virtualappliances/de-DE-resources.xml
```

477 6 OVF Descriptor

478 The OVF descriptor contains the metadata about the package and its contents. This is an extensible
479 XML document for encoding information, such as product details, virtual hardware requirements, and
480 licensing.

481 The DMTF DSP8023 schema definition file for the OVF descriptor contains the elements and attributes.
482 The OVF descriptor shall validate with the DMTF [DSP8023](#).

483 Clauses 7, 8, and 9, describe the semantics, structure, and extensibility framework of the OVF descriptor.
 484 These clauses are not a replacement for reading the schema definitions, but they complement the
 485 schema definitions.

486 The XML namespaces used in this specification are listed in Table 1. The choice of any namespace prefix
 487 is arbitrary and not semantically significant.

488 **Table 1 – XML Namespace Prefixes**

Prefix	XML Namespace
ovf	http://schemas.dmtf.org/ovf/envelope/2
ovfenv	http://schemas.dmtf.org/ovf/environment/1
rasd	http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/ CIM_ResourceAllocationSettingData.xsd
vssd	http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/ CIM_VirtualSystemSettingData.xsd
epasd	http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/ CIM_EthernetPortAllocationSettingData.xsd
sasd	http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/ CIM_StorageAllocationSettingData.xsd
cim	http://schemas.dmtf.org/wbem/wscim/1/common.xsd

489 7 Envelope Element

490 The `Envelope` element describes all metadata for the virtual machines (including virtual hardware), as
 491 well as the structure of the OVF package itself.

492 The outermost level of the envelope consists of the following parts:

- 493 • A version indication, defined by the XML namespace URIs.
- 494 • A list of file references to all external files that are part of the OVF package, defined by the
 495 `References` element and its `File` child elements. These are typically virtual disk files, ISO
 496 images, and internationalization resources.
- 497 • A metadata part, defined by section elements, as defined in clause 9.
- 498 • A description of the content, either a single virtual machine (`VirtualSystem` element) or a
 499 collection of multiple virtual machines (`VirtualSystemCollection` element).
- 500 • A specification of message resource bundles for zero or more locales, defined by a `Strings`
 501 element for each locale.

502 **EXAMPLE:** An example of the structure of an OVF descriptor with the top-level `Envelope` element follows:

```
503 <?xml version="1.0" encoding="UTF-8"?>
504 <Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
505   xmlns:vssd="http://schemas.dmtf.org/wbem/wscim/1/cim-
506   schema/2/CIM_VirtualSystemSettingData"
507   xmlns:rasd="http://schemas.dmtf.org/wbem/wscim/1/cim-
508   schema/2/CIM_ResourceAllocationSettingData"
509   xmlns:ovf="http://schemas.dmtf.org/ovf/envelope/2"
510   xmlns="http://schemas.dmtf.org/ovf/envelope/2"
511   xml:lang="en-US">
512   <References>
513     <File ovf:id="de-DE-resources.xml" ovf:size="15240"
514       ovf:href="http://mywebsite/virtualappliances/de-DE-resources.xml"/>
```



```

515     <File ovf:id="file1" ovf:href="vmdisk1.vmdk" ovf:size="180114671"/>
516     <File ovf:id="file2" ovf:href="vmdisk2.vmdk" ovf:size="4882023564"
517 ovf:chunkSize="2147483648"/>
518     <File ovf:id="file3" ovf:href="resource.iso" ovf:size="212148764"
519 ovf:compression="gzip"/>
520     <File ovf:id="icon" ovf:href="icon.png" ovf:size="1360"/>
521 </References>
522 <!-- Describes meta-information about all virtual disks in the package -->
523 <DiskSection>
524     <Info>Describes the set of virtual disks</Info>
525     <!-- Additional section content -->
526 </DiskSection>
527 <!-- Describes all networks used in the package -->
528 <NetworkSection>
529     <Info>List of logical networks used in the package</Info>
530     <!-- Additional section content -->
531 </NetworkSection>
532 <SomeSection ovf:required="false">
533     <Info>A plain-text description of the content</Info>
534     <!-- Additional section content -->
535 </SomeSection>
536 <!-- Additional sections can follow -->
537 <VirtualSystemCollection ovf:id="Some Product">
538     <!-- Additional sections including VirtualSystem or VirtualSystemCollection-->
539 </VirtualSystemCollection >
540 <Strings xml:lang="de-DE">
541     <!-- Specification of message resource bundles for de-DE locale -->
542 </Strings>
543 </Envelope>

```

544 The optional `xml:lang` attribute on the `Envelope` element shall specify the default locale for messages
545 in the descriptor. The optional `Strings` elements shall contain string resource bundles for different
546 locales. See clause 10 for more details on internationalization support.

547 7.1 File References

548 The file reference part defined by the `References` element allows a tool to easily determine the integrity
549 of an OVF package without having to parse or interpret the entire structure of the descriptor. Tools can
550 safely manipulate (for example, copy or archive) OVF packages with no risk of losing files.

551 External string resource bundle files for internationalization shall be placed first in the `References`
552 element, see clause 10 for details.

553 Each `File` element in the reference part shall be given an identifier using the `ovf:id` attribute. The
554 identifier shall be unique inside an OVF package. Each `File` element shall be specified using the
555 `ovf:href` attribute, which shall contain a URL. Relative-path references and the URL schemes "file",
556 "http", and "https" shall be supported, see [RFC1738](#) and [RFC3986](#). Other URL schemes should not
557 be used. If no URL scheme is specified, the value of the `ovf:href` attribute shall be interpreted as a
558 path name of the referenced file relative to the location of the OVF descriptor itself. The relative path
559 name shall use the syntax of relative-path references in [RFC3986](#). The referenced file shall exist. Two
560 different `File` elements shall not reference the same file with their `ovf:href` attributes.

561 The size of the referenced file may be specified using the `ovf:size` attribute. The unit of this attribute
562 shall be bytes. If present, the value of the `ovf:size` attribute should match the actual size of the
563 referenced file.

564 Each file referenced by a `File` element may be compressed using gzip (see [RFC1952](#)). When a `File`
565 element is compressed using gzip, the `ovf:compression` attribute shall be set to "gzip". Otherwise,
566 the `ovf:compression` attribute shall be set to "identity" or the entire attribute omitted. Alternatively,

567 if the href is an HTTP or HTTPS URL, then the compression may be specified by the HTTP server by
 568 using the HTTP header `Content-Encoding: gzip` (see [RFC2616](#)). Using HTTP content encoding in
 569 combination with the `ovf:compression` attribute is allowed, but in general does not improve the
 570 compression ratio. When compression is used, the `ovf:size` attribute shall specify the size of the actual
 571 compressed file.

572 Files referenced from the reference part may be split into chunks to accommodate file size restrictions on
 573 certain file systems. Chunking shall be indicated by the presence of the `ovf:chunkSize` attribute; the
 574 value of `ovf:chunkSize` shall be the size of each chunk, except the last chunk, which may be smaller.

575 When `ovf:chunkSize` is specified, the `File` element shall reference a chunk file representing a chunk
 576 of the entire file. In this case, the value of the `ovf:href` attribute specifies only a part of the URL, and
 577 the syntax for the URL resolving to the chunk file shall be as follows.

```
578 chunk-url      = href-value "." chunk-number
579 chunk-number  = 9(decimal-digit)
580 decimal-digit = "0" | "1" | "2" | "3" | "4" | "5" | "6" | "7" | "8" | "9"
```

581 The syntax is defined in ABNF notation with the exceptions listed in ANNEX A. The href-value shall be
 582 the value of the `ovf:href` attribute. The chunk-number shall be the 0-based position of the chunk
 583 starting with the value 0 and increasing with increments of 1 for each chunk.

584 If chunking is combined with compression, the entire file shall be compressed before chunking and each
 585 chunk shall be an equal slice of the compressed file, except for the last chunk which may be smaller.

586 If the OVF package has a manifest file, the file name in the manifest entries shall match the value of the
 587 `ovf:href` attribute for the file, except if the file is split into multiple chunks, in which case the `chunk-`
 588 `url` shall be used, and the manifest file shall contain an entry for each individual chunk. If chunked files
 589 are used, the manifest file may contain an entry for the entire file; and if present this digest shall also be
 590 verified.

591 EXAMPLE 1: The following example shows different types of file references:

```
592 <File ovf:id="disk1" ovf:href="disk1.vmdk"/>
593 <File ovf:id="disk2" ovf:href="disk2.vmdk" ovf:size="5368709120"
594         ovf:chunkSize="2147483648"/>
595 <File ovf:id="iso1" ovf:href="resources/image1.iso"/>
596 <File ovf:id="iso2" ovf:href="http://mywebsite/resources/image2.iso"/>
```

597 EXAMPLE 2: The following example shows manifest entries corresponding to the file references above:

```
598 SHA1(disk1.vmdk)= 3e19644ec2e806f38951789c76f43e4a0ec7e233
599 SHA1(disk2.vmdk.000000000)= 4f7158731ff434380bf217da248d47a2478e79d8
600 SHA1(disk2.vmdk.000000001)= 12849daeeaf43e7a89550384d26bd437bb8defaf
601 SHA1(disk2.vmdk.000000002)= 4cdd21424bd9eeafa4c42112876217de2ee5556d
602 SHA1(resources/image1.iso)= 72b37ff3fdd09f2a93f1b8395654649b6d06b5b3
603 SHA1(http://mywebsite/resources/image2.iso)=
604 d3c2d179011c970615c5cf10b30957d1c4c968ad
```

605 7.2 Content Element

606 Virtual machine configurations in an OVF package are represented by a `VirtualSystem` or
 607 `VirtualSystemCollection` element. These elements shall be given an identifier using the `ovf:id`
 608 attribute. Direct child elements of a `VirtualSystemCollection` shall have unique identifiers.

609 In the OVF schema, the `VirtualSystem` and `VirtualSystemCollection` elements are part of a
 610 substitution group with the `Content` element as head of the substitution group. The `Content` element is
 611 abstract and cannot be used directly. The OVF descriptor shall have one or more `Content` elements.

612 The `VirtualSystem` element describes a single virtual machine and is simply a container of section
 613 elements. These section elements describe virtual hardware, resources, and product information and are
 614 described in detail in clauses 8 and 9.

615 An example of a `VirtualSystem` element structure is as follows:

```
616 <VirtualSystem ovf:id="simple-app">
617   <Info>A virtual machine</Info>
618   <Name>Simple Appliance</Name>
619   <SomeSection>
620     <!-- Additional section content -->
621   </SomeSection>
622   <!-- Additional sections can follow -->
623 </VirtualSystem>
```

624 The `VirtualSystemCollection` element is a container of multiple `VirtualSystem` or
 625 `VirtualSystemCollection` elements. Thus, arbitrary complex configurations can be described. The
 626 section elements at the `VirtualSystemCollection` level describe appliance information, properties,
 627 resource requirements, and so on, and are described in detail in clause 9.

628 An example of a `VirtualSystemCollection` element structure is as follows:

```
629 <VirtualSystemCollection ovf:id="multi-tier-app">
630   <Info>A collection of virtual machines</Info>
631   <Name>Multi-tiered Appliance</Name>
632   <SomeSection>
633     <!-- Additional section content -->
634   </SomeSection>
635   <!-- Additional sections can follow -->
636   <VirtualSystem ovf:id="...">
637     <!-- Additional sections -->
638   </VirtualSystem>
639   <!-- Additional VirtualSystem or VirtualSystemCollection elements can follow-->
640 </VirtualSystemCollection>
```

641 All elements in the `Content` substitution group contain an `Info` element and may contain a `Name`
 642 element. The `Info` element contains a human readable description of the meaning of this entity. The
 643 `Name` element is an optional localizable display name of the content. See clause 10 for details on how to
 644 localize the `Info` and `Name` element.

645 7.3 Extensibility

646 This specification allows custom meta-data to be added to OVF descriptors in several ways:

- 647 • New section elements may be defined as part of the `Section` substitution group, and used
 648 where the OVF schemas allow sections to be present. All subtypes of `Section` contain an `Info`
 649 element that contains a human readable description of the meaning of this entity. The values of
 650 `Info` elements can be used, for example, to give meaningful warnings to users when a section is
 651 being skipped, even if the parser does not know anything about the section. See clause 10 for
 652 details on how to localize the `Info` element.
- 653 • The OVF schemas use an open content model, where all existing types may be extended at the
 654 end with additional elements. Extension points are declared in the OVF schemas with `xs:any`
 655 declarations with `namespace="##other"`.
- 656 • The OVF schemas allow additional attributes on existing types.

657 Custom extensions shall not use XML namespaces defined in this specification. This applies to both
 658 custom elements and custom attributes.

659 On custom elements, a Boolean `ovf:required` attribute specifies whether the information in the
 660 element is required for correct behavior or optional. If not specified, the `ovf:required` attribute defaults
 661 to TRUE. A consumer of an OVF package that detects an extension that is required and that it does not
 662 understand shall fail.

663 For known `Section` elements, if additional child elements that are not understood are found and the
 664 value of their `ovf:required` attribute is TRUE, the consumer of the OVF package shall interpret the
 665 entire section as one it does not understand. The check is not recursive; it applies only to the direct
 666 children of the `Section` element. This behavior ensures that older parsers reject newer OVF
 667 specifications, unless explicitly instructed not to do so.

668 On custom attributes, the information in the attribute shall not be required for correct behavior.

669 EXAMPLE 1:

```
670 <!-- Optional custom section example -->
671 <otherns:IncidentTrackingSection ovf:required="false">
672   <Info>Specifies information useful for incident tracking purposes</Info>
673   <BuildSystem>Acme Corporation Official Build System</BuildSystem>
674   <BuildNumber>102876</BuildNumber>
675   <BuildDate>10-10-2008</BuildDate>
676 </otherns:IncidentTrackingSection>
```

677 EXAMPLE 2:

```
678 <!-- Open content example (extension of existing type) -->
679 <AnnotationSection>
680   <Info>Specifies an annotation for this virtual machine</Info>
681   <Annotation>This is an example of how a future element (Author) can still be
682     parsed by older clients</Annotation>
683   <!-- AnnotationSection extended with Author element -->
684   <otherns:Author ovf:required="false">John Smith</otherns:Author>
685 </AnnotationSection>
```

686 EXAMPLE 3:

```
687 <!-- Optional custom attribute example -->
688 <Network ovf:name="VM network" otherns:desiredCapacity="1 Gbit/s">
689   <Description>The main network for VMs</Description>
690 </Network>
```

691 7.4 Conformance

692 This specification defines three conformance levels for OVF descriptors, with 1 being the highest level of
 693 conformance:

- 694 • OVF descriptor uses only sections and elements and attributes that are defined in this
 695 specification.
 696 Conformance Level: 1.
- 697 • OVF descriptor uses custom sections or elements or attributes that are not defined in this
 698 specification, and all such extensions are optional as defined in 7.3.
 699 Conformance Level: 2.
- 700 • OVF descriptor uses custom sections or elements that are not defined in this specification and at
 701 least one such extension is required as defined in 7.3. The definition of all required extensions
 702 shall be publicly available in an open and unencumbered XML Schema. The complete
 703 specification may be inclusive in the XML schema or available as a separate document.
 704 Conformance Level: 3.

705 The use of conformance level 3 limits portability and should be avoided if at all possible.

706 The conformance level is not specified directly in the OVF descriptor but shall be determined by the
 707 above rules.

708 8 Virtual Hardware Description

709 8.1 VirtualHardwareSection

710 Each VirtualSystem element may contain one or more VirtualHardwareSection elements, each of which
 711 describes the virtual hardware required by the virtual system. The virtual hardware required by a virtual
 712 machine is specified in VirtualHardwareSection elements. This specification supports abstract or
 713 incomplete hardware descriptions in which only the major devices are described. The virtualization
 714 platform may create additional virtual hardware controllers and devices, as long as the required devices
 715 listed in the descriptor are realized.

716 This virtual hardware description is based on the CIM classes CIM_VirtualSystemSettingData,
 717 CIM_ResourceAllocationSettingData, CIM_EthernetPortAllocationSettingData, and
 718 CIM_StorageAllocationSettingData. The XML representation of the CIM model is based on the
 719 WS-CIM mapping (DSP0230). Note: This means that the XML elements that belong to the class
 720 complex type should be ordered by Unicode code point (binary) order of their CIM property name
 721 identifiers.
 722

723 EXAMPLE: Example of VirtualHardwareSection:

```

724 <VirtualHardwareSection>
725   <Info>Memory = 4 GB, CPU = 1 GHz, Disk = 100 GB, 1 Ethernet nic</Info>
726   <Item>
727     <rasd:AllocationUnits>Hertz*10^9</rasd:AllocationUnits>
728     <rasd:Description>Virtual CPU</rasd:Description>
729     <rasd:ElementName>1 GHz virtual CPU</rasd:ElementName>
730     <rasd:InstanceID>1</rasd:InstanceID>
731     <rasd:Reservation>1</rasd:Reservation>
732     <rasd:ResourceType>3</rasd:ResourceType>
733     <rasd:VirtualQuantity>1</rasd:VirtualQuantity>
734     <rasd:VirtualQuantityUnit>Count</ rasd:VirtualQuantityUnit>
735   </Item>
736   <Item>
737     <rasd:AllocationUnits>byte*2^30</rasd:AllocationUnits>
738     <rasd:Description>Memory</rasd:Description>
739     <rasd:ElementName>1 GByte of memory</rasd:ElementName>
740     <rasd:InstanceID>2</rasd:InstanceID>
741     <rasd:Limit>4</rasd:Limit>
742     <rasd:Reservation>4</rasd:Reservation>
743     <rasd:ResourceType>4</rasd:ResourceType>
744   </Item>
745   <EthernetPortItem>
746     <epasd:Address>00-16-8B-DB-00-5E</epasd:Address>
747     <rasd:AllocationUnits>bit / second *2^30 </rasd:AllocationUnits> VERIFY
748 the PUnit for Gbits per second
749     <epasd:Connection>VM Network</epasd:Connection>
750     <epasd:Description>Virtual NIC</epasd:Description>
751
752     <epasd:ElementName>Ethernet Port</epasd:ElementName>
753     <epasd:InstanceID>3</epasd:InstanceID>
754     <epasd:NetworkPortProfileID>1</epasd:NetworkPortProfileID>
755     <epasd:NetworkPortProfileIDType>4</epasd:NetworkPortProfileIDType>
756     <epasd:ResourceType>10</epasd:ResourceType>
757     <epasd:VirtualQuantity>1</epasd:VirtualQuantity>
758     <epasd:VirtualQuantityUnits>Count</epasd:VirtualQuantityUnits>
759   </EthernetPortItem>
760   <StorageItem>
761     <sasd:AllocationUnits>byte*2^30</sasd:AllocationUnits>
762     <sasd:Description>Virtual Disk</sasd:Description>
763     <sasd:ElementName>100 GByte Virtual Disk</sasd:ElementName>

```

```

764         <rasd:InstanceID>4</rasd:InstanceID>
765         <rasd:Reservation>100</rasd:Reservation>
766         <rasd:ResourceType>31</rasd:ResourceType>
767         <rasd:VirtualQuantity>1</rasd:VirtualQuantity>
768         <rasd:VirtualQuantityUnit>Count</rasd:VirtualQuantityUnit>
769     </StorageItem>
770 </VirtualHardwareSection>

```

771 A `VirtualSystem` element shall have a `VirtualHardwareSection` direct child element.
 772 `VirtualHardwareSection` shall not be a direct child element of a `VirtualSystemCollection`
 773 element and of an `Envelope` element.

774 Multiple `VirtualHardwareSection` element occurrences are allowed within a single `VirtualSystem`
 775 element. The consumer of the OVF package should select the most appropriate virtual hardware
 776 description for the particular virtualization platform. A `VirtualHardwareSection` element may contain
 777 an `ovf:id` attribute which can be used to identify the element. If present the attribute value must be
 778 unique within the `VirtualSystem`.

779 The `ovf:transport` attribute specifies the types of transport mechanisms by which properties are
 780 passed to the virtual machine in an OVF environment document. This attribute supports a pluggable and
 781 extensible architecture for providing guest/platform communication mechanisms. Several transport types
 782 may be specified separated by single space character. See 9.5 for a description of properties and clause
 783 11 for a description of transport types and OVF environments.

784 A `VirtualHardwareSection` element contains sub elements that describe virtual system and virtual
 785 hardware resources (CPU, memory, network, and storage).

786 A `VirtualHardwareSection` element shall have zero or one `System` direct child element, followed by
 787 zero or more `Item` direct child elements, zero or more `EthernetPortItem` direct child elements, and
 788 zero or more `StorageItem` direct child elements.

789 The `System` element is an XML representation of the values of one or more properties of the CIM class
 790 `CIM_VirtualSystemSettingData`. The `vssd:VirtualSystemType`, a direct child element of
 791 `System` element, specifies a virtual system type identifier, which is an implementation defined string that
 792 uniquely identifies the type of the virtual system. For example, a virtual system type identifier could be
 793 `vmx-4` for VMware's fourth-generation virtual hardware or `xen-3` for Xen's third-generation virtual
 794 hardware. Zero or more virtual system type identifiers may be specified separated by single space
 795 character. In order for the OVF virtual system to be deployable on a target platform, the virtual machine
 796 on the target platform should support at least one of the virtual system types identified in the
 797 `vssd:VirtualSystemType` elements. The virtual system type identifiers specified in
 798 `vssd:VirtualSystemType` elements are expected to be matched against the values of property
 799 `VirtualSystemTypesSupported` of CIM class `CIM_VirtualSystemManagementCapabilities`.

800 The virtual hardware characteristics are described as a sequence of `Item` elements. The `Item` element
 801 is an XML representation of an instance of the CIM class `CIM_ResourceAllocationSettingData`.
 802 The element can describe all memory and CPU requirements as well as virtual hardware devices.

803 Multiple device subtypes may be specified in an `Item` element, separated by a single space character.

804 EXAMPLE:

```

805 <rasd:ResourceSubType>buslogic lsilogic</rasd:ResourceSubType>

```

806 The network hardware characteristics are described as a sequence of `EthernetPortItem` elements.
 807 The `EthernetPortItem` element is an XML representation of the values of one or more properties of
 808 the CIM class `CIM_EthernetPortAllocationSettingData`.

809 The storage hardware characteristics are described as a sequence of `StorageItem` elements. The
 810 `StorageItem` element is an XML representation of the values of one or more properties of the CIM class
 811 `CIM_StorageAllocationSettingData`.

812 8.2 Extensibility

813 The optional `ovf:required` attribute on the `Item`, `EthernetPortItem`, or `StorageItem`
 814 element specifies whether the realization of the element (for example, a CD-ROM or USB controller) is
 815 required for correct behavior of the guest software. If not specified, `ovf:required` defaults to `TRUE`.

816 On child elements of the `Item`, `EthernetPortItem`, or `StorageItem` element, the optional
 817 Boolean attribute `ovf:required` shall be interpreted, even though these elements are in a different
 818 RASD WS-CIM namespace. A tool parsing an `Item` element should act according to Table 2.

819 **Table 2 – Actions for Child Elements with `ovf:required` Attribute**

Child Element	<code>ovf:required</code> Attribute Value	Action
Known	TRUE or not specified	Shall interpret <code>Item</code> , <code>EthernetPortItem</code> , or <code>StorageItem</code>
Known	FALSE	Shall interpret <code>Item</code> , <code>EthernetPortItem</code> , or <code>StorageItem</code>
Unknown	TRUE or not specified	Shall fail <code>Item</code> , <code>EthernetPortItem</code> , or <code>StorageItem</code>
Unknown	FALSE	Shall ignore Child Element

820 8.3 Virtual Hardware Elements

821 The element type of the `Item` element in a `VirtualHardwareSection` element is
 822 `CIM_ResourceAllocationSettingData_Type` as defined in [http://schemas.dmtf.org/wbem/wscim/1/cim-](http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/CIM_ResourceAllocationSettingData.xsd)
 823 [schema/2/CIM_ResourceAllocationSettingData.xsd](http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/CIM_ResourceAllocationSettingData.xsd).

824 The child elements of `Item` represent the values of one or more properties exposed by the
 825 `CIM_ResourceAllocationSettingData` class. They have the semantics of defined settings as
 826 defined in [DSP1041](#), any profiles derived from [DSP1041](#) for specific resource types, and this document.

827 EXAMPLE: The following example shows a description of memory size:

```
828 <Item>
829   <rasd:AllocationUnits>byte * 2^20</rasd:AllocationUnits>
830   <rasd:Description>Memory Size</rasd:Description>
831   <rasd:ElementName>256 MB of memory</rasd:ElementName>
832   <rasd:InstanceID>2</rasd:InstanceID>
833   <rasd:ResourceType>4</rasd:ResourceType>
834   <rasd:VirtualQuantity>256</rasd:VirtualQuantity>
835 </Item>
```

836 The element type of the `EthernetPortItem` element in a `VirtualHardwareSection` element is
 837 `CIM_EthernetPortAllocationSettingData_Type` as defined in [http://schemas.dmtf.org/wbem/wscim/1/cim-](http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/CIM_EthernetPortAllocationSettingData.xsd)
 838 [schema/2/CIM_EthernetPortAllocationSettingData.xsd](http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/CIM_EthernetPortAllocationSettingData.xsd).

839 The child elements represent the values of one or more properties exposed by the
 840 `CIM_EthernetPortAllocationSettingData` class. They have the semantics of defined settings as

841 defined in [DSP1050](#), any profiles derived from [DSP1050](#) for specific Ethernet port resource types, and
842 this document.

843 EXAMPLE: The following example shows a description of a virtual Ethernet adapter:

```
844 <EthernetPortItem>
845   <epasd:Address>00-16-8B-DB-00-5E</epasd:Address>
846   <epasd:Connection>VM Network</epasd:Connection>
847   <epasd:Description>Virtual NIC</epasd:Description>
848   <epasd:ElementName>Ethernet Port 1</epasd:ElementName>
849   <epasd:InstanceID>3</epasd:InstanceID>
850   <epasd:NetworkPortProfileID>1</epasd:NetworkPortProfileID>
851   <epasd:NetworkPortProfileIDType>4</epasd:NetworkPortProfileIDType>
852   <epasd:VirtualQuantityUnits>1</epasd:VirtualQuantityUnits>
853 </EthernetPortItem>
```

854 The element type of the `StorageItem` element in a `VirtualHardwareSection` element is
855 `CIM_StorageAllocationSettingData_Type` as defined in [http://schemas.dmtf.org/wbem/wscim/1/cim-
856 schema/2/CIM_StorageAllocationSettingData.xsd](http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/CIM_StorageAllocationSettingData.xsd).

857 The child elements represent the values of one or more properties exposed by the
858 `CIM_StorageAllocationSettingData` class. They have the semantics of defined settings as defined
859 in [DSP1047](#), any profiles derived from [DSP1047](#) for specific storage resource types, and this document.

860 EXAMPLE: The following example shows a description of a virtual storage:

```
861 <StorageItem>
862   <sasd:AllocationUnits>byte*2^30</sasd:AllocationUnits>
863   <sasd:Description>Virtual Disk</sasd:Description>
864   <sasd:ElementName>100 GByte Virtual Disk</sasd:ElementName>
865   <sasd:InstanceID>4</sasd:InstanceID>
866   <sasd:Reservation>100</sasd:Reservation>
867   <sasd:ResourceType>31</sasd:ResourceType>
868   <sasd:VirtualQuantity>1</sasd:VirtualQuantity>
869 </StorageItem>
```

870 The `Description` element is used to provide additional metadata about the `Item`, `EthernetPortItem`, or
871 `StorageItem` element itself. This element enables a consumer of the OVF package to provide descriptive
872 information about all items, including items that were unknown at the time the application was written.

873 The `Caption`, `Description` and `ElementName` elements are localizable using the `ovf:msgid`
874 attribute from the OVF envelope namespace. See clause 10 for more details on internationalization
875 support.

876 The optional `ovf:configuration` attribute contains a list of configuration names. See 9.8 on
877 deployment options for semantics of this attribute. The optional `ovf:bound` attribute is used to specify
878 ranges; see 8.4.

879 Devices such as disks, CD-ROMs, and networks need a backing from the deployment platform. The
880 requirements on a backing are either specified using the `HostResource` or the `Connection` element.

881 For an Ethernet adapter, a logical network name is specified in the `Connection` element. Ethernet
882 adapters that refer to the same logical network name within an OVF package shall be deployed on the
883 same network.

884 The `HostResource` element is used to refer to resources included in the OVF descriptor as well as
885 logical devices on the deployment platform. Values for `HostResource` elements referring to resources
886 included in the OVF descriptor are formatted as URIs as specified in Table 3.

887

Table 3 – HostResource Element

Content	Description
ovf:/file/<id>	A reference to a file in the OVF, as specified in the References section. <id> shall be the value of the <code>ovf:id</code> attribute of the <code>File</code> element being referenced.
ovf:/disk/<id>	A reference to a virtual disk, as specified in the <code>DiskSection</code> or <code>SharedDiskSection</code> . <id> shall be the value of the <code>ovf:diskId</code> attribute of the <code>Disk</code> element being referenced.

888 If no backing is specified for a device that requires a backing, the deployment platform shall make an
 889 appropriate choice, for example, by prompting the user. More than one backing for a device shall not be
 890 specified.

891 Table 4 gives a brief overview on how elements from `rasd`, `epasd`, and `sasd` namespaces are used to
 892 describe virtual devices and controllers.

893

Table 4 – Elements for Virtual Devices and Controllers

Element	Usage
Description	A human-readable description of the meaning of the information. For example, "Specifies the memory size of the virtual machine".
ElementName	A human-readable description of the content. For example, "256MB memory".
InstanceID	A unique instance ID of the element within the section.
HostResource	Abstractly specifies how a device shall connect to a resource on the deployment platform. Not all devices need a backing. See Table 3.
ResourceType OtherResourceType ResourceSubtype	Specifies the kind of device that is being described.
AutomaticAllocation	For devices that are connectable, such as floppies, CD-ROMs, and Ethernet adaptors, this element specifies whether the device should be connected at power on.
Parent	The InstanceID of the parent controller (if any).
Connection	For an Ethernet adapter, this specifies the abstract network connection name for the virtual machine. All Ethernet adapters that specify the same abstract network connection name within an OVF package shall be deployed on the same network. The abstract network connection name shall be listed in the <code>NetworkSection</code> at the outermost envelope level.
Address	Device specific. For an Ethernet adapter, this specifies the MAC address.
AddressOnParent	For a device, this specifies its location on the controller.
AllocationUnits	Specifies the unit of allocation used. For example, "byte * 2^20".
VirtualQuantity	Specifies the quantity of resources presented. For example, "256".
Reservation	Specifies the minimum quantity of resources guaranteed to be available.
Limit	Specifies the maximum quantity of resources that are granted.
Weight	Specifies a relative priority for this allocation in relation to other allocations.

894 Only fields directly related to describing devices are mentioned. Refer to the CIM MOF for a complete
 895 description of all fields, each field corresponds to the identically named property in the
 896 `CIM_ResourceAllocationSettingData` class or a class derived from it.

8.4 Ranges on Elements

The optional `ovf:bound` attribute may be used to specify ranges for the `Item` elements. A range has a minimum, normal, and maximum value, denoted by `min`, `normal`, and `max`, where `min <= normal <= max`. The default values for `min` and `max` are those specified for `normal`.

A platform deploying an OVF package should start with the normal value and adjust the value within the range for ongoing performance tuning and validation.

For the `Item`, `EthernetPortItem`, and `StorageItem` elements in `VirtualHardwareSection` and `ResourceAllocationSection` elements, the following additional semantics are defined:

- Each `Item`, `EthernetPortItem`, or `StorageItem` element has an optional `ovf:bound` attribute. This value may be specified as `min`, `max`, or `normal`. The value defaults to `normal`. If the attribute is not specified or is specified as `normal`, then the item shall be interpreted as being part of the regular virtual hardware or resource allocation description.
- If the `ovf:bound` value is specified as either `min` or `max`, the item is used to specify the upper or lower bound for one or more values for a given `InstanceID`. Such an item is called a range marker.

The semantics of range markers are as follows:

- `InstanceID` and `ResourceType` shall be specified, and the `ResourceType` shall match other `Item` elements with the same `InstanceID`.
- More than one `min` range marker nor more than one `max` range marker for a given RASD, EPASD, or SASD (identified with `InstanceID`) shall not be specified..
- An `Item`, `EthernetPortItem`, or `StorageItem` element with a range marker shall have a corresponding `Item`, `EthernetPortItem`, or `StorageItem` element without a range marker, that is, an `Item`, `EthernetPortItem`, and `StorageItem` element with no `ovf:bound` attribute or `ovf:bound` attribute with value `normal`. This corresponding item specifies the default value.
- For an `Item`, `EthernetPortItem`, and `StorageItem` element where only a `min` range marker is specified, the `max` value is unbounded upwards within the set of valid values for the property.
- For an `Item`, `EthernetPortItem`, and `StorageItem` where only a `max` range marker is specified, the `min` value is unbounded downwards within the set of valid values for the property.
- The default value shall be inside the range.
- Non-integer elements shall not be used in the range markers for RASD, EPASD, or SASD.

EXAMPLE: The following example shows the use of range markers:

```

930 <VirtualHardwareSection>
931   <Info>...</Info>
932   <Item>
933     <rasd:AllocationUnits>byte * 2^20</rasd:AllocationUnits>
934     <rasd:ElementName>512 MB memory size</rasd:ElementName>
935     <rasd:InstanceID>0</rasd:InstanceID>
936     <rasd:ResourceType>4</rasd:ResourceType>
937     <rasd:VirtualQuantity>512</rasd:VirtualQuantity>
938   </Item>
939   <Item ovf:bound="min">
940     <rasd:AllocationUnits>byte * 2^20</rasd:AllocationUnits>
941     <rasd:ElementName>384 MB minimum memory size</rasd:ElementName>
942     <rasd:InstanceID>0</rasd:InstanceID>

```

```
943     <rasd:Reservation>384</rasd:Reservation>
944     <rasd:ResourceType>4</rasd:ResourceType>
945   </Item>
946   <Item ovf:bound="max">
947     <rasd:AllocationUnits>byte * 2^20</rasd:AllocationUnits>
948     <rasd:ElementName>1024 MB maximum memory size</rasd:ElementName>
949     <rasd:InstanceID>0</rasd:InstanceID>
950     <rasd:Reservation>1024</rasd:Reservation>
951     <rasd:ResourceType>4</rasd:ResourceType>
952   </Item>
953 </VirtualHardwareSection>
954
```

955 **9 Core Metadata Sections in version 2**956 Table 5 shows the core metadata sections that are defined in the `ovf` namespace.957 **Table 5 – Core Metadata Sections in version 2**

Section	Locations	Multiplicity
<code>DiskSection</code> Describes meta-information about all virtual disks in the package	Envelope	Zero or one
<code>NetworkSection</code> Describes logical networks used in the package	Envelope	Zero or one
<code>ResourceAllocationSection</code> Specifies reservations, limits, and shares on a given resource, such as memory or CPU for a virtual machine collection	VirtualSystemCollection	Zero or one
<code>AnnotationSection</code> Specifies a free-form annotation on an entity	VirtualSystem VirtualSystemCollection	Zero or one
<code>ProductSection</code> Specifies product-information for a package, such as product name and version, along with a set of properties that can be configured	VirtualSystem VirtualSystemCollection	Zero or more
<code>EulaSection</code> Specifies a license agreement for the software in the package	VirtualSystem VirtualSystemCollection	Zero or more
<code>StartupSection</code> Specifies how a virtual machine collection is powered on	VirtualSystemCollection	Zero or one
<code>DeploymentOptionSection</code> Specifies a discrete set of intended resource requirements	Envelope	Zero or one
<code>OperatingSystemSection</code> Specifies the installed guest operating system of a virtual machine	VirtualSystem	Zero or one
<code>InstallSection</code> Specifies that the virtual machine needs to be initially booted to install and configure the software	VirtualSystem	Zero or one
<code>EnvironmentFilesSection</code> Specifies additional files from an OVF package to be included in the OVF environment	VirtualSystem	Zero or one
<code>BootDeviceSection</code> Specifies boot device order to be used by a virtual machine	VirtualSystem	Zero or more
<code>SharedDiskSection</code> Specifies virtual disks shared by more than one VirtualSystems at runtime	Envelope	Zero or one
<code>ScaleOutSection</code> Specifies that a VirtualSystemCollection contain a set of children that are homogeneous with respect to a prototype	VirtualSystemCollection	Zero or more
<code>PlacementGroupSection</code> Specifies a placement policy for a group of VirtualSystems or VirtualSystemCollections	Envelope	Zero or more

Section	Locations	Multiplicity
PlacementSection Specifies membership of a particular placement policy group	VirtualSystem VirtualSystemCollection	Zero or one
EncryptionSection Specifies encryption scheme for encrypting parts of an OVF descriptor or files that it refers to.	Envelope	Zero or one

958 The following subclauses describe the semantics of the core sections and provide some examples. The
 959 sections are used in several places of an OVF envelope; the description of each section defines where it
 960 may be used. See the OVF schema for a detailed specification of all attributes and elements.

961 In the OVF schema, all sections are part of a substitution group with the `Section` element as head of the
 962 substitution group. The `Section` element is abstract and cannot be used directly.

963 9.1 DiskSection

964 A `DiskSection` describes meta-information about virtual disks in the OVF package. Virtual disks and
 965 their metadata are described outside the virtual hardware to facilitate sharing between virtual machines
 966 within an OVF package. Virtual disks in `DiskSection` can be referenced by multiple virtual machines,
 967 but seen from the guest software each virtual machine get individual private disks. Any level of sharing
 968 done at runtime is deployment platform specific and not visible to the guest software. See clause 9.13 for
 969 details on how to configure sharing of virtual disk at runtime with concurrent access.

970 EXAMPLE: The following example shows a description of virtual disks:

```

971 <DiskSection>
972   <Info>Describes the set of virtual disks</Info>
973   <Disk ovf:diskId="vmdisk1" ovf:fileRef="file1" ovf:capacity="8589934592"
974     ovf:populatedSize="3549324972"
975     ovf:format=
976       "http://www.vmware.com/interfaces/specifications/vmdk.html#sparse">
977   </Disk>
978   <Disk ovf:diskId="vmdisk2" ovf:capacity="536870912">
979   </Disk>
980   <Disk ovf:diskId="vmdisk3" ovf:capacity="${disk.size}"
981     ovf:capacityAllocationUnits="byte * 2^30"
982   </Disk>
983 </DiskSection>
  
```

984 `DiskSection` is a valid section at the outermost envelope level only.

985 Each virtual disk represented by a `Disk` element shall be given an identifier using the `ovf:diskId`
 986 attribute; the identifier shall be unique within the `DiskSection`.

987 The capacity of a virtual disk shall be specified by the `ovf:capacity` attribute with an `xs:long` integer
 988 value. The default unit of allocation shall be bytes. The optional string attribute
 989 `ovf:capacityAllocationUnits` may be used to specify a particular unit of allocation. Values for
 990 `ovf:capacityAllocationUnits` shall match the format for programmatic units defined in [DSP0004](#)
 991 with the restriction that the base unit shall be "byte".

992 The `ovf:fileRef` attribute denotes the virtual disk content by identifying an existing `File` element in
 993 the `References` element, the `File` element is identified by matching its `ovf:id` attribute value with the
 994 `ovf:fileRef` attribute value. Omitting the `ovf:fileRef` attribute shall indicate an empty disk. In this
 995 case, the disk shall be created and the entire disk content zeroed at installation time. The guest software
 996 will typically format empty disks in some file system format.

997 The format URI (see 5.2) of a non-empty virtual disk shall be specified by the `ovf:format` attribute.

1098 Different `Disk` elements shall not contain `ovf:fileRef` attributes with identical values. `Disk` elements
 1099 shall be ordered such that they identify any `File` elements in the same order as these are defined in the
 1100 `References` element.

1101 For empty disks, rather than specifying a fixed virtual disk capacity, the capacity for an empty disk may be
 1102 given using an OVF property, for example `ovf:capacity="{disk.size}"`. The OVF property shall
 1103 resolve to an `xs:long` integer value. See 9.5 for a description of OVF properties. The
 1104 `ovf:capacityAllocationUnits` attribute is useful when using OVF properties because a user may
 1105 be prompted and can then enter disk sizing information in ,for example, gigabytes.

1106 For non-empty disks, the actual used size of the disk may optionally be specified using the
 1107 `ovf:populatedSize` attribute. The unit of this attribute shall be bytes. The `ovf:populatedSize`
 1108 attribute may be an estimate of used disk size but shall not be larger than `ovf:capacity`.

1109 In `VirtualHardwareSection`, virtual disk devices may have a `rasd:HostResource` element
 1110 referring to a `Disk` element in `DiskSection`; see 8.3. The virtual disk capacity shall be defined by the
 1111 `ovf:capacity` attribute on the `Disk` element. If a `rasd:VirtualQuantity` element is specified along
 1112 with the `rasd:HostResource` element, the virtual quantity value shall not be considered and may have
 1113 any value.

1114 OVF allows a disk image to be represented as a set of modified blocks in comparison to a parent image.
 1115 The use of parent disks can often significantly reduce the size of an OVF package if it contains multiple
 1116 disks with similar content, such as a common base operating system. Actual sharing of disk blocks at
 1117 runtime is optional and deployment platform specific and shall not be visible to the guest software.

1118 For the `Disk` element, a parent disk may optionally be specified using the `ovf:parentRef` attribute,
 1119 which shall contain a valid `ovf:diskId` reference to a different `Disk` element. If a disk block does not
 1120 exist locally, lookup for that disk block then occurs in the parent disk. In `DiskSection`, parent `Disk`
 1121 elements shall occur before child `Disk` elements that refer to them. Similarly, in `References` element,
 1122 the `File` elements referred from these `Disk` elements shall respect the same ordering. The ordering
 1123 restriction ensures that in an OVA archive, parent disks always occur before child disks, making it
 1124 possible for a tool to consume the archive in a streaming mode, see also clause 5.3.

1025 9.2 NetworkSection

1026 The `NetworkSection` element shall list all logical networks used in the OVF package.

```

1027 <NetworkSection>
1028   <Info>List of logical networks used in the package</Info>
1029   <Network ovf:name="VM Network">
1030     <Description>The network that the service will be available on</Description>
1031     <NetworkPortProfile>
1032       <Item>
1033         <epasd:AllocationUnits>GigaBits per Second</epasd:AllocationUnits>
1034         <epasd:ElementName>Network Port Profile 1</epasd:ElementName>
1035         <epasd:InstanceID>1</epasd:InstanceID>
1036         <epasd:NetworkPortProfileID>1</epasd:NetworkPortProfileID>
1037         <epasd:NetworkPortProfileIDType>4</epasd:NetworkPortProfileIDType>
1038         <epasd:Reservation>1</epasd:Reservation>
1039       </Item>
1040     </NetworkPortProfile>
1041   </Network>
1042 </NetworkSection>

```

1043 `NetworkSection` is a valid element at the outermost envelope level. A `Network` element is a child
 1044 element of `NetworkSection`. Each `Network` element in the `NetworkSection` shall be given a unique
 1045 name using the `ovf:name` attribute. The name shall be unique within an `ovf` envelope.

1046 All networks referred to from `Connection` elements in all `VirtualHardwareSection` elements shall
1047 be defined in the `NetworkSection`.

1048 Starting with version 2.0 of this specification, each logical network may contain a set of networking
1049 attributes that should be applied when mapping the logical network at deployment time to a physical or
1050 virtual network. Networking attributes are specified by embedding or referencing zero or more instances
1051 of network port profile as specified by `NetworkPortProfile` or `NetworkPortProfileURI` child
1052 element of the `Network` element.

1053 The `NetworkPortProfile` child element of the `Network` element defines the contents of a network
1054 port profile. The `NetworkPortProfileURI` child element of the `Network` element defines the
1055 reference to a network port profile.

1056 Examples of using the DSP8049 and EPASD are in ANNEX D.

1057 9.3 ResourceAllocationSection

1058 The `ResourceAllocationSection` element describes all resource allocation requirements of a
1059 `VirtualSystemCollection` entity. These resource allocations shall be performed when deploying the
1060 OVF package.

```
1061 <ResourceAllocationSection>
1062   <Info>Defines reservations for CPU and memory for the collection of VMs</Info>
1063   <Item>
1064     <rasd:AllocationUnits>byte * 2^20</rasd:AllocationUnits>
1065     <rasd:ElementName>300 MB reservation</rasd:ElementName>
1066     <rasd:InstanceID>0</rasd:InstanceID>
1067     <rasd:Reservation>300</rasd:Reservation>
1068     <rasd:ResourceType>4</rasd:ResourceType>
1069   </Item>
1070   <Item ovf:configuration="..." ovf:bound="...">
1071     <rasd:AllocationUnits>hertz * 10^6</rasd:AllocationUnits>
1072     <rasd:ElementName>500 MHz reservation</rasd:ElementName>
1073     <rasd:InstanceID>0</rasd:InstanceID>
1074     <rasd:Reservation>500</rasd:Reservation>
1075     <rasd:ResourceType>3</rasd:ResourceType>
1076   </Item>
1077   <EthernetPortItem>
1078     <epasd:Address>00-16-8B-DB-00-5E</epasd:Address>
1079     <epasd:Connection>VM Network</epasd:Connection>
1080     <epasd:Description>Virtual NIC</epasd:Description>
1081     <epasd:ElementName>Ethernet Port 1</epasd:ElementName>
1082     <epasd:InstanceID>3</epasd:InstanceID>
1083     <epasd:NetworkPortProfileID>1</epasd:NetworkPortProfileID>
1084     <epasd:NetworkPortProfileIDType>4</epasd:NetworkPortProfileIDType>
1085     <epasd:VirtualQuantityUnits>1</epasd:VirtualQuantityUnits>
1086   </EthernetPortItem>
1087   <StorageItem>
1088     <sasd:AllocationUnits>byte*2^30</sasd:AllocationUnits>
1089     <sasd:Description>Virtual Disk</sasd:Description>
1090     <sasd:ElementName>100 GByte Virtual Disk</sasd:ElementName>
1091     <sasd:InstanceID>4</sasd:InstanceID>
1092     <sasd:Reservation>100</sasd:Reservation>
1093     <sasd:ResourceType>31</sasd:ResourceType>
1094     <sasd:VirtualQuantity>1</sasd:VirtualQuantity>
1095   </StorageItem>
1096 </ResourceAllocationSection>
```

1097 `ResourceAllocationSection` is a valid element for a `VirtualSystemCollection` entity.

1098 The optional `ovf:configuration` attribute contains a list of configuration names. See 9.8 on
1099 deployment options for semantics of this attribute.

1100 The optional `ovf:bound` attribute contains a value of `min`, `max`, or `normal`. See 8.4 for semantics of this
1101 attribute.

1102 9.4 AnnotationSection

1103 The `AnnotationSection` element is a user-defined annotation on an entity. Such annotations may be
1104 displayed when deploying the OVF package.

```
1105 <AnnotationSection>
1106   <Info>An annotation on this service. It can be ignored</Info>
1107   <Annotation>Contact customer support if you have any problems</Annotation>
1108 </AnnotationSection >
```

1109 `AnnotationSection` is a valid element for a `VirtualSystem` and a `VirtualSystemCollection`
1110 entity.

1111 See clause 10 for details on how to localize the `Annotation` element.

1112 9.5 ProductSection

1113 The `ProductSection` element specifies product-information for an appliance, such as product name,
1114 version, and vendor.

```
1115 <ProductSection ovf:class="com.mycrm.myservice" ovf:instance="1">
1116   <Info>Describes product information for the service</Info>
1117   <Product>MyCRM Enterprise</Product>
1118   <Vendor>MyCRM Corporation</Vendor>
1119   <Version>4.5</Version>
1120   <FullVersion>4.5-b4523</FullVersion>
1121   <ProductUrl>http://www.mycrm.com/enterprise</ProductUrl>
1122   <VendorUrl>http://www.mycrm.com</VendorUrl>
1123   <Icon ovf:height="32" ovf:width="32" ovf:mimeType="image/png" ovf:fileRef="icon">
1124     <Category>Email properties</Category>
1125     <Property ovf:key="admin.email" ovf:type="string" ovf:userConfigurable="true">
1126       <Label>Admin email</Label>
1127       <Description>Email address of administrator</Description>
1128     </Property>
1129     <Category>Admin properties</Category>
1130     <Property ovf:key="app_log" ovf:type="string" ovf:value="low"
1131     ovf:userConfigurable="true">
1132       <Description>Loglevel for the service</Description>
1133     </Property>
1134     <Property ovf:key="app_isSecondary" ovf:value="false" ovf:type="boolean">
1135       <Description>Cluster setup for application server</Description>
1136     </Property>
1137     <Property ovf:key="app_ip" ovf:type="string" ovf:value="{appserver-vm}">
1138       <Description>IP address of the application server VM</Description>
1139     </Property>
1140 </ProductSection>
```

1141 The optional `Product` element specifies the name of the product, while the optional `Vendor` element
1142 specifies the name of the product vendor. The optional `Version` element specifies the product version in
1143 short form, while the optional `FullVersion` element describes the product version in long form. The
1144 optional `ProductUrl` element specifies a URL which shall resolve to a human readable description of
1145 the product, while the optional `VendorUrl` specifies a URL which shall resolve to a human readable
1146 description of the vendor.

- 1147 The optional `AppUrl` element specifies a URL resolving to the deployed product instance. The optional
1148 `Icon` element specifies display icons for the product.
- 1149 The `Property` elements specify application-level customization parameters and are particularly relevant
1150 to appliances that need to be customized during deployment with specific settings such as network
1151 identity, the IP addresses of DNS servers, gateways, and others.
- 1152 The `ProductSection` is a valid section for a `VirtualSystem` and a `VirtualSystemCollection` entity.
- 1153 The `Property` elements may be grouped by using `Category` elements. The set of `Property` elements
1154 grouped by a `Category` element is the sequence of `Property` elements following the `Category`
1155 element, until but not including an element that is not a `Property` element. For OVF packages
1156 containing a large number of `Property` elements, this may provide a simpler installation experience.
1157 Similarly, each `Property` element may have a short label defined by its `Label` child element in addition
1158 to a description defined by its `Description` child element. See clause 10 for details on how to localize
1159 the `Category` element and the `Description` and `Label` child elements of the `Property` element.
- 1160 Each `Property` element in a `ProductSection` shall be given an identifier that is unique within the
1161 `ProductSection` using the `ovf:key` attribute. The `ovf:key` attribute shall not contain the period
1162 character (".") or the colon character (":").
- 1163 Each `Property` element in a `ProductSection` shall be given a type using the `ovf:type` attribute and
1164 optionally type qualifiers using the `ovf:qualifiers` attribute. Valid types are listed in Table 6, and valid
1165 qualifiers are listed in Table 7.
- 1166 The optional attribute `ovf:value` is used to provide a default value for a property. One or more optional
1167 `Value` elements may be used to define alternative default values for different configurations, as defined
1168 in 9.8.
- 1169 The optional attribute `ovf:userConfigurable` determines whether the property value is configurable
1170 during the installation phase. If `ovf:userConfigurable` is `FALSE` or omitted, the `ovf:value` attribute
1171 specifies the value to be used for that customization parameter during installation. If
1172 `ovf:userConfigurable` is `TRUE`, the `ovf:value` attribute specifies a default value for that
1173 customization parameter, which may be changed during installation.
- 1174 A simple OVF implementation such as a command-line installer typically uses default values for
1175 properties and does not prompt even though `ovf:userConfigurable` is set to `TRUE`. To force
1176 prompting at startup time, omitting the `ovf:value` attribute is sufficient for integer types, because the
1177 empty string is not a valid integer value. For string types, prompting may be forced by adding a qualifier
1178 requiring a non-empty string, see Table 7.
- 1179 The optional Boolean attribute `ovf:password` indicates that the property value may contain sensitive
1180 information. The default value is `FALSE`. OVF implementations prompting for property values are advised
1181 to obscure these values when `ovf:password` is set to `TRUE`. This is similar to HTML text input of type
1182 `password`. Note that this mechanism affords limited security protection only. Although sensitive values
1183 are masked from casual observers, default values in the OVF descriptor and assigned values in the OVF
1184 environment are still passed in clear text.
- 1185 Zero or more `ProductSections` may be specified within a `VirtualSystem` or
1186 `VirtualSystemCollection`. Typically, a `ProductSection` corresponds to a particular software
1187 product that is installed. Each product section at the same entity level shall have a unique `ovf:class`
1188 and `ovf:instance` attribute pair. For the common case where only a single `ProductSection` is used,
1189 the `ovf:class` and `ovf:instance` attributes are optional and default to the empty string. The `ovf:class`
1190 and `ovf:instance` attributes shall not contain the colon character (":"). The `ovf:class` property should be
1191 used to uniquely identify the software product using the reverse domain name convention. Examples of
1192 values are `com.vmware.tools` and `org.apache.tomcat`. If multiple instances of the same product

1193 are installed, the `ovf:instance` attribute shall be used to identify the different instances. If only one
1194 instance of a product is installed, the `ovf:instance` attribute should not be set.

1195 Property elements are exposed to the guest software through the OVF environment, as described in
1196 clause 11. The value of the `ovfenv:key` attribute of a `Property` element exposed in the OVF
1197 environment shall be constructed from the value of the `ovf:key` attribute of the corresponding
1198 `Property` element defined in a `ProductSection` entity of an OVF descriptor as follows:

1199 `key-value-env = [class-value "."] key-value-prod ["." instance-value]`

1200 The syntax definition above use ABNF with the exceptions listed in ANNEX A, where:

- 1201 • `class-value` is the value of the `ovf:class` attribute of the `Property` element defined in the
1202 `ProductSection` entity. The production `[class-value "."]` shall be present if and only if
1203 `class-value` is not the empty string.
- 1204 • `key-value-prod` is the value of the `ovf:key` attribute of the `Property` element defined in the
1205 `ProductSection` entity.
- 1206 • `instance-value` is the value of the `ovf:instance` attribute of the `Property` element defined in
1207 the `ProductSection` entity. The production `["." instance-value]` shall be present if and only
1208 if `instance-value` is not the empty string.

1209 **EXAMPLE:** The following OVF environment example shows how properties can be propagated to the guest
1210 software:

1211 `<Property ovf:key="com.vmware.tools.logLevel" ovf:value="none"/>`
1212 `<Property ovf:key="org.apache.tomcat.logLevel.1" ovf:value="debug"/>`
1213 `<Property ovf:key="org.apache.tomcat.logLevel.2" ovf:value="normal"/>`

1214 The consumer of an OVF package should prompt for properties where `ovf:userConfigurable` is
1215 `TRUE`. These properties may be defined in multiple `ProductSections` as well as in sub-entities in the
1216 OVF package.

1217 If a `ProductSection` exists, then the first `ProductSection` entity defined in the top-level `Content`
1218 element of a package shall define summary information that describes the entire package. After
1219 installation, a consumer of the OVF package could choose to make this information available as an
1220 instance of the `CIM_Product` class.

1221 `Property` elements specified on a `VirtualSystemCollection` are also seen by its immediate
1222 children (see clause 11). Children may refer to the properties of a parent `VirtualSystemCollection`
1223 using macros on the form `#{name}` as value for `ovf:value` attributes.

1224 Table 6 lists the valid types for properties. These are a subset of CIM intrinsic types defined in [DSP0004](#),
1225 which also define the value space and format for each intrinsic type. Each `Property` element shall
1226 specify a type using the `ovf:type` attribute.

1227 **Table 6 – Property Types**

Type	Description
<code>uint8</code>	Unsigned 8-bit integer
<code>sint8</code>	Signed 8-bit integer
<code>uint16</code>	Unsigned 16-bit integer
<code>sint16</code>	Signed 16-bit integer
<code>uint32</code>	Unsigned 32-bit integer
<code>sint32</code>	Signed 32-bit integer

Type	Description
uint64	Unsigned 64-bit integer
sint64	Signed 64-bit integer
String	String
Boolean	Boolean
real32	IEEE 4-byte floating point
real64	IEEE 8-byte floating point

1228 Table 7 lists the supported CIM type qualifiers as defined in [DSP0004](#). Each `Property` element may
 1229 optionally specify type qualifiers using the `ovf:qualifiers` attribute with multiple qualifiers separated
 1230 by commas; see production `qualifierList` in ANNEX A “MOF Syntax Grammar Description” in
 1231 [DSP0004](#).

1232 **Table 7 – Property Qualifiers**

Type	Description
String	MinLen (min) MaxLen (max) ValueMap{...}
uint8 sint8 uint16 sint16 uint32 sint32 uint64 sint64	ValueMap{...}

1233 **9.6 EulaSection**

1234 A `EulaSection` contains the legal terms for using its parent `Content` element. This license shall be
 1235 shown and accepted during deployment of an OVF package. Multiple `EulaSections` may be present in
 1236 an OVF. If unattended installations are allowed, all embedded license sections are implicitly accepted.

```

1237 <EulaSection>
1238   <Info>Licensing agreement</Info>
1239   <License>
1240 Lorem ipsum dolor sit amet, ligula suspendisse nulla pretium, rhoncus tempor placerat
1241 fermentum, enim integer ad vestibulum volutpat. Nisl rhoncus turpis est, vel elit,
1242 congue wisi enim nunc ultricies sit, magna tincidunt. Maecenas aliquam maecenas ligula
1243 nostra, accumsan taciti. Sociis mauris in integer, a dolor netus non dui aliquet,
1244 sagittis felis sodales, dolor sociis mauris, vel eu libero cras. Interdum at. Eget
1245 habitasse elementum est, ipsum purus pede porttitor class, ut adipiscing, aliquet sed
1246 auctor, imperdiet arcu per diam dapibus libero duis. Enim eros in vel, volutpat nec
1247 pellentesque leo, scelerisque.
1248   </License>
1249 </EulaSection>
    
```

1250 The `EulaSection` is a valid section for a `VirtualSystem` and a `VirtualSystemCollection` entity.

1251 See clause 10 for details on how to localize the `License` element.

1252 See also clause 10 for description of storing EULA license contents in an external file without any XML
1253 header or footer. This allows inclusion of standard license or copyright text files in unaltered form.

1254 9.7 StartupSection

1255 The StartupSection specifies how a virtual machine collection is powered on and off.

```
1256 <StartupSection>
1257   <Item ovf:id="vm1" ovf:order="0" ovf:startDelay="30" ovf:stopDelay="0"
1258     ovf:startAction="powerOn" ovf:waitingForGuest="true"
1259   ovf:stopAction="powerOff"/>
1260   <Item ovf:id="teamA" ovf:order="0"/>
1261   <Item ovf:id="vm2" ovf:order="1" ovf:startDelay="0" ovf:stopDelay="20"
1262     ovf:startAction="powerOn" ovf:stopAction="guestShutdown"/>
1263 </StartupSection>
```

1264 Each Content element that is a direct child of a VirtualSystemCollection may have a
1265 corresponding Item element in the StartupSection entity of the VirtualSystemCollection entity.
1266 Note that Item elements may correspond to both VirtualSystem and VirtualSystemCollection
1267 entities. When a start or stop action is performed on a VirtualSystemCollection entity, the
1268 respective actions on the Item elements of its StartupSection entity are invoked in the specified
1269 order. Whenever an Item element corresponds to a (nested) VirtualSystemCollection entity, the
1270 actions on the Item elements of its StartupSection entity shall be invoked before the action on the
1271 Item element corresponding to that VirtualSystemCollection entity is invoked (i.e., depth-first
1272 traversal).

1273 The following required attributes on Item are supported for a VirtualSystem and
1274 VirtualSystemCollection:

- 1275 • ovf:id shall match the value of the ovf:id attribute of a Content element which is a direct
1276 child of this VirtualSystemCollection. That Content element describes the virtual
1277 machine or virtual machine collection to which the actions defined in the Item element apply.
- 1278 • ovf:order specifies the startup order using non-negative integer values. If the ovf:order
1279 ="0" then the order is not specified. If the ovf:order is non-zero then the of execution of the
1280 start action shall be the numerical ascending order of the values. The Items with same order
1281 identifier may be started concurrently.

1282 The order of execution of the stop action should be the numerical descending order of the
1283 values. In implementation specific scenarios the order of execution of the stop action may be
1284 non-descending.

1285 The following optional attributes on Item are supported for a VirtualSystem.

- 1286 • ovf:startDelay specifies a delay in seconds to wait until proceeding to the next order in the
1287 start sequence. The default value is 0.
- 1288 • ovf:waitingForGuest enables the platform to resume the startup sequence after the guest
1289 software has reported it is ready. The interpretation of this is deployment platform specific. The
1290 default value is FALSE.
- 1291 • ovf:startAction specifies the start action to use. Valid values are powerOn and none. The
1292 default value is powerOn.

- 1293 • `ovf:stopDelay` specifies a delay in seconds to wait until proceeding to the previous order in
1294 the stop sequence. The default value is 0.
- 1295 • `ovf:stopAction` specifies the stop action to use. Valid values are `powerOff`,
1296 `guestShutdown`, and `none`. The interpretation of `guestShutdown` is deployment platform
1297 specific. The default value is `powerOff`.

1298 If the `StartupSection` is not specified then an `ovf:order="0"` is implied.

1299 9.8 DeploymentOptionSection

1300 The `DeploymentOptionSection` specifies a discrete set of intended resource configurations. The
1301 author of an OVF package can include sizing metadata for different configurations. A consumer of the
1302 OVF shall select a configuration, for example, by prompting the user. The selected configuration shall be
1303 available in the OVF environment file, enabling the guest software to adapt to the selected configuration.
1304 See clause 11.

1305 The `DeploymentOptionSection` specifies an ID, label, and description for each configuration.

```
1306 <DeploymentOptionSection>
1307   <Configuration ovf:id="minimal">
1308     <Label>Minimal</Label>
1309     <Description>Some description</Description>
1310   </Configuration>
1311   <Configuration ovf:id="normal" ovf:default="true">
1312     <Label>Typical</Label>
1313     <Description>Some description</Description>
1314   </Configuration>
1315   <!-- Additional configurations -->
1316 </DeploymentOptionSection>
```

1317 The `DeploymentOptionSection` has the following semantics:

- 1318 • If present, the `DeploymentOptionSection` is valid only at the envelope level, and only one
1319 section shall be specified in an OVF descriptor.
- 1320 • The discrete set of configurations is described with `Configuration` elements, which shall
1321 have identifiers specified by the `ovf:id` attribute that are unique in the package.
- 1322 • A default `Configuration` element may be specified with the optional `ovf:default` attribute.
1323 If no default is specified, the first element in the list is the default. Specifying more than one
1324 element as the default is invalid.
- 1325 • The `Label` and `Description` elements are localizable using the `ovf:msgid` attribute. See
1326 clause 10 for more details on internationalization support.

1327 Configurations may be used to control resources for virtual hardware and for virtual machine collections.
1328 `Item`, `EthernetPortItem`, and `StorageItem` elements in `VirtualHardwareSection` elements
1329 describe resources for `VirtualSystem` entities, while `Item`, `EthernetPortItem`, and `StorageItem`
1330 elements in `ResourceAllocationSection` elements describe resources for virtual machine
1331 collections. For these two `Item`, `EthernetPortItem`, or `StorageItem` types, the following
1332 additional semantics are defined:

- 1333 • Each `Item`, `EthernetPortItem`, and `StorageItem` has an optional
1334 `ovf:configuration` attribute, containing a list of configurations separated by a single space
1335 character. If not specified, the item shall be selected for any configuration. If specified, the item
1336 shall be selected only if the chosen configuration ID is in the list. A configuration attribute shall
1337 not contain an ID that is not specified in the `DeploymentOptionSection`.
- 1338 • Within a single `VirtualHardwareSection` or `ResourceAllocationSection`, multiple
1339 `Item`, `EthernetPortItem`, and `StorageItem` elements are allowed to refer to the same
1340 `InstanceID`. A single combined `Item`, `EthernetPortItem`, or `StorageItem` for the
1341 given `InstanceID` shall be constructed by picking up the child elements of each `Item`,
1342 `EthernetPortItem`, or `StorageItem` element, with child elements of a former `Item`,
1343 `EthernetPortItem`, or `StorageItem` element in the OVF descriptor not being picked up
1344 if there is a like-named child element in a latter `Item`, `EthernetPortItem`, or
1345 `StorageItem` element. Any attributes specified on child elements of `Item`,
1346 `EthernetPortItem`, or `StorageItem` elements that are not picked up that way, are not
1347 part of the combined `Item`, `EthernetPortItem`, or `StorageItem` element.
- 1348 • All `Item`, `EthernetPortItem`, `StorageItem` elements shall specify `ResourceType`, and
1349 `Item`, `EthernetPortItem`, and `StorageItem` elements with the same `InstanceID` shall
1350 agree on `ResourceType`.

1351 EXAMPLE 1: The following example shows a `VirtualHardwareSection`:

```

1352 <VirtualHardwareSection>
1353   <Info>...</Info>
1354   <Item>
1355     <rasd:AllocationUnits>byte * 2^20</rasd:AllocationUnits>
1356     <rasd:ElementName>512 MB memory size and 256 MB
1357 reservation</rasd:ElementName>
1358     <rasd:InstanceID>0</rasd:InstanceID>
1359     <rasd:Reservation>256</rasd:Reservation>
1360     <rasd:ResourceType>4</rasd:ResourceType>
1361     <rasd:VirtualQuantity>512</rasd:VirtualQuantity>
1362   </Item>
1363   ...
1364   <Item ovf:configuration="big">
1365     <rasd:AllocationUnits>byte * 2^20</rasd:AllocationUnits>
1366     <rasd:ElementName>1024 MB memory size and 512 MB
1367 reservation</rasd:ElementName>
1368     <rasd:InstanceID>0</rasd:InstanceID>
1369     <rasd:Reservation>512</rasd:Reservation>
1370     <rasd:ResourceType>4</rasd:ResourceType>
1371     <rasd:VirtualQuantity>1024</rasd:VirtualQuantity>
1372   </Item>
1373 </VirtualHardwareSection>

```

1374 Note that the attributes `ovf:configuration` and `ovf:bound` on `Item` may be used in combination to
1375 provide very flexible configuration options.

1376 Configurations can further be used to control default values for properties and whether properties are
1377 user configurable. For `Property` elements inside a `ProductSection`, the following additional semantic
1378 is defined:

- 1379 • It is possible to specify alternative default property values for different configurations in a
1380 DeploymentOptionSection. In addition to a Label and Description element, each
1381 Property element may optionally contain Value elements. The Value element shall have
1382 an ovf:value attribute specifying the alternative default and an ovf:configuration
1383 attribute specifying the configuration in which this new default value should be used. Multiple
1384 Value elements shall not refer to the same configuration.
- 1385 • Starting with version 2.0 of this specification, a Property element may optionally have an
1386 ovf:configuration attribute specifying the configuration in which this property should be
1387 user configurable. The value of ovf:userConfigurable is implicitly set to FALSE for all
1388 other configurations, in which case the default value of the property may not be changed
1389 during installation.

1390 EXAMPLE 2: The following shows an example ProductSection:

```
1391 <ProductSection>
1392   <Property ovf:key="app.adminEmail" ovf:type="string" ovf:userConfigurable="true"
1393     ovf:configuration="standard">
1394     <Label>Admin email</Label>
1395     <Description>Email address of service administrator</Description>
1396   </Property>
1397   <Property ovf:key="app.log" ovf:type="string" ovf:value="low"
1398     ovf:userConfigurable="true">
1399     <Label>Loglevel</Label>
1400     <Description>Loglevel for the service</Description>
1401     <Value ovf:value="none" ovf:configuration="minimal">
1402   </Property>
1403 </ProductSection>
```

1404 In the example above, the app.adminEmail property is only user configurable in the standard
1405 configuration, while the default value for the app.log property is changed from low to none in the
1406 minimal configuration.

1407 9.9 OperatingSystemSection

1408 An OperatingSystemSection specifies the operating system installed on a virtual machine.

```
1409 <OperatingSystemSection ovf:id="76">
1410   <Info>Specifies the operating system installed</Info>
1411   <Description>Microsoft Windows Server 2008</Description>
1412 </OperatingSystemSection>
```

1413 The values for ovf:id should be taken from the ValueMap of the CIM_OperatingSystem.OsType
1414 property. The description should be taken from the corresponding Values of the
1415 CIM_OperatingSystem.OsType property.

1416 The OperatingSystemSection is a valid section for a VirtualSystem entity only.

1417 9.10 InstallSection

1418 The InstallSection, if specified, indicates that the virtual machine needs to be booted once in order
1419 to install and/or configure the guest software. The guest software is expected to access the OVF
1420 environment during that boot, and to shut down after having completed the installation and/or
1421 configuration of the software, powering off the guest.

1422 If the InstallSection is not specified, this indicates that the virtual machine does not need to be
1423 powered on to complete installation of guest software.

```
1424 <InstallSection ovf:initialBootStopDelay="300">
1425   <Info>Specifies that the virtual machine needs to be booted once after having
```

```

1426 created the guest software in order to install and/or configure the software
1427 </Info>
1428 </InstallSection>

```

1429 `InstallSection` is a valid section for a `VirtualSystem` entity only.

1430 The optional `ovf:initialBootStopDelay` attribute specifies a delay in seconds to wait for the virtual machine to power off. If not set, the implementation shall wait for the virtual machine to power off by itself.
 1431 If the delay expires and the virtual machine has not powered off, the consumer of the OVF package shall
 1432 indicate a failure.
 1433

1434 An `ovf:initialBootStopDelay` attribute value of zero indicates that the boot stop delay is not
 1435 specified.

1436 Note that the guest software in the virtual machine can do multiple reboots before powering off.

1437 Several VMs in a virtual machine collection may have an `InstallSection` defined, in which case the
 1438 above step is done for each VM, potentially concurrently.

1439 9.11 EnvironmentFilesSection

1440 Clause 11 describes how the OVF environment file is used to deliver runtime customization parameters to
 1441 the guest operating system. In version 1 of this specification, the OVF environment file is the only file
 1442 delivered to the guest operating system outside of the virtual disk structure. In order to provide additional
 1443 deployment time customizations, the `EnvironmentFilesSection` enables the OVF package authors
 1444 to specify additional files in the OVF package, outside of the virtual disks, that also is provided to the
 1445 guest operating system at runtime via a transport.

1446 This enables increased flexibility in image customization outside of virtual disk capture, allowing OVF
 1447 package authors to customize solutions by combining existing virtual disks without modifying them.

1448 For each additional file provided to the guest, the `EnvironmentFilesSection` shall contain a `File`
 1449 element with required attributes `ovf:fileRef` and `ovf:path`. The `ovf:fileRef` attribute shall denote
 1450 the actual content by identifying an existing `File` element in the `References` element, the `File`
 1451 element is identified by matching its `ovf:id` attribute value with the `ovf:fileRef` attribute value. The
 1452 `ovf:path` attribute denotes the relative location on the transport where this file will be placed, using the
 1453 syntax of relative-path references in [RFC3986](#).

1454 The referenced `File` element in the `References` element identify the content using one of the URL
 1455 schemes "file", "http", or "https". For the "file" scheme, the content is static and included in
 1456 the OVF package. For the "http" and "https" schemes, the content shall be downloaded prior to the
 1457 initial boot of the virtual system.

1458 The `iso` transport shall support this mechanism, see clause 11.2 for details. For this transport, the root
 1459 location relative to `ovf:path` values shall be directory `ovffiles` contained in the root directory of the
 1460 ISO image. The guest software can access the information using standard guest operating system tools.

1461 Other custom transport may support this mechanism. Custom transports will need to specify how to
 1462 access multiple data sources from a root location.

1463 EXAMPLE:

```

1464 <Envelope>
1465   <References>
1466     ...
1467     <File ovf:id="config" ovf:href="config.xml" ovf:size="4332"/>
1468     <File ovf:id="resources" ovf:href="http://mywebsite/resources/resources.zip"/>
1469   </References>
1470   ...

```



```

1471 <VirtualSystem ovf:id="...">
1472   ...
1473   <ovf:EnvironmentFilesSection ovf:required="false" ovf:transport="iso">
1474     <Info>Config files to be included in OVF environment</Info>
1475     <ovf:File ovf:fileRef="config" ovf:path="setup/cfg.xml"/>
1476     <ovf:File ovf:fileRef="resources" ovf:path="setup/resources.zip"/>
1477   </ovf:EnvironmentFilesSection>
1478   ...
1479 </VirtualSystem>
1480   ...
1481 </Envelope>

```

1482 In the example above, the file `config.xml` in the OVF package will be copied to the OVF environment
 1483 ISO image and be accessible to the guest software in location `/ovffiles/setup/cfg.xml`, while the
 1484 file `resources.zip` will be accessible in location `/ovffiles/setup/resources.zip`.

1485 9.12 BootDeviceSection

1486 Individual virtual machines will generally use the default device boot order provided by the virtualization
 1487 platform's virtual BIOS. The `BootDeviceSection` allows the OVF package author to specify particular
 1488 boot configurations and boot order settings. This enables booting from non-default devices such as a NIC
 1489 using PXE, a USB device or a secondary disk. Moreover there could be multiple boot configurations with
 1490 different boot orders. For example, a virtual disk may be need to be patched before it is bootable and a
 1491 patch ISO image could be included in the OVF package.

1492 The Common Information Model (CIM) defines artifacts to deal with boot order use cases prevalent in the
 1493 industry for BIOSes found in desktops and servers. The boot configuration is defined by the class
 1494 `CIM_BootConfigSetting` which in turn contains one or more `CIM_BootSourceSetting` classes as
 1495 defined in the WS-CIM schema. Each class representing the boot source in turn has either the specific
 1496 device or a "device type" such as disk or CD/DVD as a boot source.

1497 In the context of this specification, the `InstanceID` field of `CIM_BootSourceSetting` is used for
 1498 identifying a specific device as the boot source. The `InstanceID` field of the device as specified in the
 1499 `Item` description of the device in the `VirtualHardwareSection` is used to specify the device as a
 1500 boot source. In case the source is desired to be a device type, the `StructuredBootString` field is
 1501 used to denote the type of device with values defined by the CIM boot control profile. When a boot source
 1502 is a device type, the deployment platform should try all the devices of the specified type.

1503 In the example below, the Pre-Install configuration specifies the boot source as a specific device
 1504 (network), while the Post-Install configuration specifies a device type (hard disk).

1505 EXAMPLE:

```

1506 <Envelope>
1507 ...
1508 <VirtualSystem ovf:id="...">
1509 ...
1510 <ovf:BootDeviceSection>
1511 <Info>Boot device order specification</Info>
1512 <bootc:CIM_BootConfigSetting>
1513 <bootc:Caption>Pre-Install</bootc:Caption>
1514 <bootc:Description>Boot Sequence for fixup of disk</bootc:Description>
1515 <boots:CIM_BootSourceSetting>
1516 <boots:Caption>Fix-up DVD on the network</boots:Caption>
1517 <boots:InstanceID>3</boots:InstanceID> <!-- Network device-->
1518 </boots:CIM_BootSourceSetting>
1519 <boots:CIM_BootSourceSetting>
1520 <boots:Caption>Boot virtual disk</boots:Caption>
1521 <boots:StructuredBootString>CIM:Hard-Disk</boots:StructuredBootString>
1522 </boots:CIM_BootSourceSetting>
1523 </bootc:CIM_BootConfigSetting>
1524 </ovf:BootDeviceSection>
1525 ...
1526 </VirtualSystem>
1527 </Envelope>

```

1528 9.13 SharedDiskSection

1529 The existing `DiskSection` in clause 9.1 describes virtual disks in the OVF package. Virtual disks in the
 1530 `DiskSection` can be referenced by multiple virtual machines, but seen from the guest software each
 1531 virtual machine gets individual private disks. Any level of sharing done at runtime is deployment platform
 1532 specific and not visible to the guest software.

1533 Certain applications such as clustered databases rely on multiple virtual machines sharing the same
 1534 virtual disk at runtime. `SharedDiskSection` allows the OVF package author to specify `Disk` elements
 1535 shared by more than one `VirtualSystem` at runtime, these could be virtual disks backing by an external
 1536 `File` reference, or empty virtual disks without backing. It is recommended that the guest software use
 1537 cluster-aware file system technology to be able to handle concurrent access.

1538 EXAMPLE:

```

1539 <ovf:SharedDiskSection>
1540 <Info>Describes the set of virtual disks shared between VMs</Info>
1541 <ovf:SharedDisk ovf:diskId="datadisk" ovf:fileRef="data"
1542 <ovf:capacity="8589934592" ovf:populatedSize="3549324972"
1543 <ovf:format=
1544 "http://www.vmware.com/interfaces/specifications/vmdk.html#sparse"/>
1545 <ovf:SharedDisk ovf:diskId="transientdisk" ovf:capacity="536870912"/>
1546 </ovf:SharedDiskSection>

```

1547 `SharedDiskSection` is a valid section at the outermost envelope level only.

1548 Each virtual disk is represented by a `SharedDisk` element that shall be given an identifier using the
 1549 `ovf:diskId` attribute; the identifier shall be unique within the combined content of `DiskSection` and
 1550 `SharedDiskSection`. The `SharedDisk` element has the same structure as the `Disk` element in
 1551 `DiskSection`, with the addition of an optional boolean attribute `ovf:readOnly` stating whether shared
 1552 disk access is read-write or read-only.

1553 Shared virtual disks are referenced from virtual hardware using the `HostResource` element as described
 1554 in clause 8.3.

1555 It is optional for the virtualization platform to support `SharedDiskSection`. The platform should give an
 1556 appropriate error message based on the value of the `ovf:required` attribute on the
 1557 `SharedDiskSection` element.

1558 9.14 ScaleOutSection

1559 The number of `VirtualSystems` and `VirtualSystemCollections` contained in an OVF package is generally
 1560 fixed and determined by the structure inside the `Envelope` element. The `ScaleOutSection` allows a
 1561 `VirtualSystemCollection` to contain a set of children that are homogeneous with respect to a prototypical
 1562 `VirtualSystem` or `VirtualSystemCollection`. The `ScaleOutSection` shall cause the deployment platform
 1563 to replicate the prototype a number of times, thus allowing the number of instantiated virtual machines to
 1564 be configured dynamically at deployment time.

1565 EXAMPLE:

```
1566 <VirtualSystemCollection ovf:id="web-tier">
1567   ...
1568   <ovf:ScaleOutSection ovf:id="web-server">
1569     <Info>Web tier</Info>
1570     <ovf:Description>Number of web server instances in web tier</ovf:Description>
1571     <ovf:InstanceCount ovf:default="4" ovf:minimum="2" ovf:maximum="8"/>
1572   </ovf:ScaleOutSection>
1573   ...
1574   <VirtualSystem ovf:id="web-server">
1575     <Info>Prototype web server</Info>
1576     ...
1577   </VirtualSystem>
1578 </VirtualSystemCollection>
```

1579 In the example above, the deployment platform creates a web tier containing between two and eight web
 1580 server virtual machine instances, with a default instance count of four. The deployment platform makes
 1581 an appropriate choice (e.g., by prompting the user). Assuming three replicas were created, the OVF
 1582 environment available to the guest software in the first replica has the following content structure:

1583 EXAMPLE:

```
1584 <Environment ... ovfenv:id="web-server-1">
1585   ...
1586   <Entity ovfenv:id="web-server-2">
1587     ...
1588   </Entity>
1589   <Entity ovfenv:id="web-server-3">
1590     ...
1591   </Entity>
1592 </Environment>
```

1593 This mechanism enables dynamic scaling of virtual machine instances at deployment time. Scaling at
 1594 runtime is not within the scope of this specification.

1595 The `ScaleOutSection` is a valid section inside `VirtualSystemCollection` only.

1596 The `ovf:id` attribute on `ScaleOutSection` identifies the `VirtualSystem` or `VirtualSystemCollection`
 1597 prototype to be replicated.

1598 For the `InstanceCount` element, the `ovf:minimum` and `ovf:maximum` attribute values shall be non-
 1599 negative integers and `ovf:minimum` shall be less than or equal to the value of `ovf:maximum`. The
 1600 `ovf:minimum` value may be zero in which case the `VirtualSystemCollection` may contain zero instances
 1601 of the prototype. If the `ovf:minimum` attribute is not present, it shall be assumed to have a value of one.
 1602 If the `ovf:maximum` attribute is not present, it shall be assumed to have a value of unbounded. The
 1603 `ovf:default` attribute is required and shall contain a value within the range defined by `ovf:minimum`
 1604 and `ovf:maximum`.

1605 Each replicated instance shall be assigned a unique `ovf:id` value within the `VirtualSystemCollection`.
 1606 The unique `ovf:id` value shall be constructed from the prototype `ovf:id` value with a sequence
 1607 number appended as follows:

```
1608 replica-ovf-id = prototype-ovf-id "-" decimal-number
1609 decimal-number = decimal-digit | (decimal-digit decimal-number)
1610 decimal-digit = "0" | "1" | "2" | "3" | "4" | "5" | "6" | "7" | "8" | "9"
```

1611 The syntax definitions above use ABNF with the exceptions listed in ANNEX A. The first replica shall
 1612 have sequence number one and following sequence numbers shall be incremented by one for each
 1613 replica. Note that after deployment, no `VirtualSystem` will have the prototype `ovf:id` value itself.

1614 If the prototype being replicated has a starting order in the `StartupSection`, all replicas shall share this
 1615 value. It is not possible to specify a particular starting sequence among replicas.

1616 Property values for Property elements in the prototype are prompted for once per replica created. If the
 1617 OVF package author requires a property value to be shared among instances, that Property may be
 1618 declared at the containing `VirtualSystemCollection` level and referenced by replicas as described in
 1619 clause 9.5.

1620 Configurations from the `DeploymentOptionSection` may be used to control values for `InstanceCount`. The
 1621 `InstanceCount` element may have an `ovf:configuration` attribute specifying the configuration in
 1622 which this element should be used. Multiple elements shall not refer to the same configuration, and a
 1623 configuration attribute shall not contain an `ovf:id` value that is not specified in the
 1624 `DeploymentOptionSection`.

1625 EXAMPLE:

```
1626 <VirtualSystemCollection ovf:id="web-tier">
1627   ...
1628   <DeploymentOptionSection>
1629     <Info>Deployment size options</Info>
1630     <Configuration ovf:id="minimal">
1631       <Label>Minimal</Label>
1632       <Description>Minimal deployment scenario</Description>
1633     </Configuration>
1634     <Configuration ovf:id="common" ovf:default="true">
1635       <Label>Typical</Label>
1636       <Description>Common deployment scenario</Description>
1637     </Configuration>
1638     ...
1639   </DeploymentOptionSection>
1640   ...
1641   <ovf:ScaleOutSection ovf:id="web-server">
1642     <Info>Web tier</Info>
1643     <ovf:Description>Number of web server instances in web tier</ovf:Description>
1644     <ovf:InstanceCount ovf:default="4"/>
1645     <ovf:InstanceCount ovf:default="1" ovf:configuration="minimal"/>
1646   </ovf:ScaleOutSection>
1647   ...
1648 </VirtualSystemCollection>
```

1649 In the example above, the default replica count is four, unless the minimal deployment scenario is
 1650 chosen, in which case the default is one.

1651 9.15 PlacementGroupSection and PlacementSection

1652 Certain types of applications require the ability to specify that two or more `VirtualSystems` should be
 1653 deployed closely together since they rely on very fast communication or a common hardware dependency
 1654 such as a reliable communication link. Other types of applications require the ability to specify that two or

1655 more VirtualSystems should be deployed apart due to high-availability or disaster recovery
1656 considerations.

1657 PlacementGroupSection allow an OVF package author to define a placement policy for a group of
1658 VirtualSystems, while PlacementSection allow the author to annotate elements with membership of a
1659 particular placement policy group.

1660 Zero or more PlacementGroupSections may be declared at the Envelope level, while
1661 PlacementSection may be declared at the VirtualSystem or VirtualSystemCollection level. Declaring a
1662 VirtualSystemCollection member of a placement policy group applies transitively to all child VirtualSystem
1663 and child Virtual System Collections elements. The ovf:id attribute on PlacementGroupSection is
1664 used to identify the particular placement policy; the attribute value shall be unique within the OVF
1665 package. Placement policy group membership is specified using the ovf:group attribute on
1666 PlacementSection; the attribute value shall match the value of an ovf:id attribute on a
1667 PlacementGroupSection.

1668 This version of the specification defines the placement policies "affinity" and "availability",
1669 specified with the required ovf:policy attribute on PlacementGroupSection.

1670 Placement policy "affinity" describe that VirtualSystems should be placed as closely together as
1671 possible. The deployment platform should attempt to keep these virtual machines located as adjacently
1672 as possible, typically on the same physical host or with fast network connectivity between hosts.

1673 Placement policy "availability" describe that VirtualSystems should be placed separately. The
1674 deployment platform should attempt to keep these virtual machines located apart, typically on the
1675 different physical hosts.

1676 EXAMPLE:

```
1677 <Envelope ...>
1678   ...
1679   <ovf:PlacementGroupSection ovf:id="web" ovf:policy="availability">
1680     <Info>Placement policy for group of VMs</Info>
1681     <ovf:Description>Placement policy for web tier</ovf:Description>
1682   </ovf:PlacementGroupSection>
1683   ...
1684   <VirtualSystemCollection ovf:id="web-tier">
1685     ...
1686     <ovf:ScaleOutSection ovf:id="web-node">
1687       <Info>Web tier</Info>
1688       ...
1689     </ovf:ScaleOutSection>
1690     ...
1691     <VirtualSystem ovf:id="web-node">
1692       <Info>Web server</Info>
1693       ...
1694       <ovf:PlacementSection ovf:group="web">
1695         <Info>Placement policy group reference</Info>
1696       </ovf:PlacementSection>
1697       ...
1698     </VirtualSystem>
1699   </VirtualSystemCollection>
1700 </Envelope>
```

1701 In the example above, all virtual machines in the compute tier should be placed separately for high
1702 availability. This example also use the ScaleOutSection defined in clause 9.14, in which case each
1703 replica get the policy assigned.

1704 **9.16 Encryption Section**

1705 For licensing and other reasons it is desirable to have an encryption scheme enabling free exchange of
 1706 OVF appliances while ensuring that only the intended parties can use them. The XML Encryption Syntax
 1707 and Processing standard is utilized to encrypt either the files in the reference section or any parts of the
 1708 XML markup of an OVF document.

1709 The various aspects of OVF encryption are as shown below:

- 1710 1. block encryption
 1711 The OVF document author shall utilize block encryption algorithms as specified in the XML
 1712 encryption 1.1 documents (ref) for this purpose.
- 1713 2. key derivation
 1714 The OVF author may use the appropriate key for this purpose. If the key is derived using a
 1715 passphrase then the author shall use one of the key derivations specified in the XML Encryption
 1716 1.1 standard.
- 1717 3. key transport.
 1718 If the encryption key is embedded in the OVF document, the specified key transport mechanisms
 1719 shall be used.

1720 This specification defines a new section called the EncryptionSection as a focal point for the encryption
 1721 functionality. This new section provides a single location for placing the encryption algorithm related
 1722 markup and the corresponding reference list to point to the OVF content that has been encrypted.

1723 Note that depending on which parts of the OVF markup has been encrypted, an OVF descriptor may not
 1724 validate against the OVF schemas until decrypted.

1725 Below is an example of an OVF encryption section with encryption methods utilized in the OVF
 1726 document, and the corresponding reference list pointing to the items that have been encrypted.

1727 **EXAMPLE:**

```

1728 <ovf:EncryptionSection>
1729 <!-- This section contains two different methods of encryption and the corresponding
1730 backpointers to the data that is encrypted ->
1731 <!-- Method#1: Pass phrase based Key derivation ->
1732 <!-- The following derived key block defines PBKDF2 and the corresponding back
1733 pointers to the encrypted data elements -->
1734 <!-- Use a salt value "ovfpassword" and iteration count of 4096 --->
1735 <xenc11:DerivedKey>
1736 <xenc11:KeyDerivationMethod
1737 Algorithm="http://www.rsasecurity.com/rsalabs/pkcs/schemas/pkcs-5#pbkdf2"/>
1738 <pkcs-5:PBKDF2-params>
1739 <Salt>
1740 <Specified>ovfpassword</Specified>
1741 </Salt>
1742 <IterationCount>4096</IterationCount>
1743 <KeyLength>16</KeyLength>
1744 <PRF Algorithm="http://www.w3.org/2001/04/xmldsig-more#hmac-sha256"/>
1745 </pkcs-5:PBKDF2-params>
1746 ...
1747 <!-- The ReferenceList element below contains references to the file Ref-109.vhd via
1748 the URI syntax which is specified by XML Encryption.
1749 --->
1750 <xenc:ReferenceList>
1751 <xenc:DataReference URI="#first.vhd" />
1752 <xenc:DataReference URI=... />
1753 <xenc:DataReference URI=... />
1754 </xenc:ReferenceList>
1755 </xenc11:DerivedKey>
1756 <!-- Method#2: The following example illustrates use of a symmetric key
  
```

```

1757 transported using the public key within a certificate ->
1758 <xenc:EncryptedKey>
1759     <xenc:EncryptionMethod Algorithm="http://www.w3.org/2001/04/xmlenc#rsa-
1760 1_5"/>
1761     <ds:KeyInfo xmlns:ds='http://www.w3.org/2000/09/xmldsig#'
1762         <ds:X509Data>
1763             <ds:X509Certificate> ... </ds:X509Certificate>
1764         </ds:X509Data>
1765     </ds:KeyInfo>
1766     <xenc:CipherData>
1767     <xenc:CipherValue> ... </xenc:CipherValue>
1768     </xenc:CipherData>
1769 <!-- The ReferenceList element below contains reference #second-xml-fragment" to the
1770 XML fragment that has been encrypted using the above method --->
1771 <xenc:ReferenceList>
1772     <xenc:DataReference URI='#second-xml-fragment' />
1773     <xenc:DataReference URI='...' />
1774     <xenc:DataReference URI='...' />
1775 </xenc:ReferenceList>
1776 </xenc:EncryptedKey>
1777 </ovf:EncryptionSection>

```

1778 Below is an example of the encrypted file which is referenced in the EncryptionSection above using
 1779 URI='Ref-109.vhd' syntax.

1780 **EXAMPLE:**

```

1781 <ovf:References>
1782 <ovf:File ovf:id="Xen:9cb10691-4012-4aeb-970c-3d47a906bfff/0b13bdba-3761-8622-22fc-
1783 2e252ed9ce14" ovf:href="Ref-109.vhd">
1784 <!-- the encrypted file referenced by the package is enclosed by an EncryptedData with
1785 a CipherReference to the actual encrypted data. The EncryptionSection in this example
1786 has a back pointer to it under the PBKDF2 algorithm via Id="first.vhd". This tells the
1787 decrypter how to decrypt the file -->
1788 <xenc:EncryptedData Id="first.vhd" Type='http://www.w3.org/2001/04/xmlenc#Element' >
1789     <xenc:EncryptionMethod
1790 Algorithm="http://www.w3.org/2001/04/xmlenc#aes128-cbc" />
1791     <xenc:CipherData>
1792     <xenc:CipherReference URI='Ref-109.vhd' />
1793     </xenc:CipherData>
1794 </xenc:EncryptedData>
1795 </ovf:File>
1796 </ovf:References>

```

1797 Below is an example of the encrypted OVF markup which is referenced in the EncryptionSection above
 1798 using URI='#second-xml-fragment' syntax.

1799 **EXAMPLE:**

```

1800 <!-- the EncryptedData element below encompasses encrypted xml from the original
1801 document. It is provided with the Id "first-xml-fragment" which allows it to be
1802 referenced from the EncryptionSection. -->
1803 <xenc:EncryptedData Type=http://www.w3.org/2001/04/xmlenc#Element Id="second-xml-
1804 fragment">
1805 <!-- Each EncryptedData specifies its own encryption method. -->
1806 <xenc:EncryptionMethod Algorithm=http://www.w3.org/2001/04/xmlenc#aes128-cbc/>
1807 <xenc:CipherData>
1808 <!-- Encrypted content --->
1809 <xenc:CipherValue>DEADBEEF</xenc:CipherValue>
1810 </xenc:CipherData>
1811 </xenc:EncryptedData>

```

1812 10 Internationalization

1813 The following elements support localizable messages using the optional `ovf:msgid` attribute:

- 1814 • Info element on Content
- 1815 • Name element on Content
- 1816 • Info element on Section
- 1817 • Annotation element on AnnotationSection
- 1818 • License element on EulaSection
- 1819 • Description element on NetworkSection
- 1820 • Description element on OperatingSystemSection
- 1821 • Description, Product, Vendor, Label, and Category elements on ProductSection
- 1822 • Description and Label elements on Property
- 1823 • Description and Label elements on DeploymentOptionSection
- 1824 • ElementName, Caption and Description subelements on the System element in
- 1825 VirtualHardwareSection
- 1826 • ElementName, Caption and Description subelements on Item elements in
- 1827 VirtualHardwareSection
- 1828 • ElementName, Caption and Description subelements on Item elements in
- 1829 ResourceAllocationSection

1830 The `ovf:msgid` attribute contains an identifier that refers to a message that may have different values in
1831 different locales.

1832 EXAMPLE 1:

```
1833 <Info ovf:msgid="info.text">Default info.text value if no locale is set or no locale
1834 match</Info>
1835 <License ovf:msgid="license.tomcat-6_0"/> <!-- No default message -->
```

1836 The `xml:lang` attribute on the `Envelope` element shall specify the default locale for messages in the
1837 descriptor. The attribute is optional with a default value of "en-US".

1838 10.1 Internal Resource Bundles

1839 Message resource bundles can be internal or external to the OVF descriptor. Internal resource bundles
1840 are represented as `Strings` elements at the end of the `Envelope` element.

1841 EXAMPLE 2:

```
1842 <ovf:Envelope xml:lang="en-US">
1843   ...
1844   ... sections and content here ...
1845   ...
1846   <Info msgid="info.os">Operating System</Info>
1847   ...
1848   <Strings xml:lang="da-DA">
1849     <Msg ovf:msgid="info.os">Operativsystem</Msg>
1850     ...
1851   </Strings>
1852   <Strings xml:lang="de-DE">
1853     <Msg ovf:msgid="info.os">Betriebssystem</Msg>
1854     ...
```



```
1855     </Strings>
1856 </ovf:Envelope>
```

1857 10.2 External Resource Bundles

1858 External resource bundles shall be listed first in the `References` section and referred to from `Strings`
 1859 elements. An external message bundle follows the same schema as the embedded one. Exactly one
 1860 `Strings` element shall be present in an external message bundle, and that `Strings` element may not
 1861 have an `ovf:fileRef` attribute specified.

1862 EXAMPLE 3:

```
1863 <ovf:Envelope xml:lang="en-US">
1864   <References>
1865     ...
1866     <File ovf:id="it-it-resources" ovf:href="resources/it-it-bundle.msg"/>
1867   </References>
1868   ... sections and content here ...
1869   ...
1870   <Strings xml:lang="it-IT" ovf:fileRef="it-it-resources"/>
1871   ...
1872 </ovf:Envelope>
```

1873 EXAMPLE 4: Example content of external resources/it-it-bundle.msg file, which is referenced in previous example:

```
1874 <Strings
1875   xmlns:ovf="http://schemas.dmtf.org/ovf/envelope/1"
1876   xmlns="http://schemas.dmtf.org/ovf/envelope/1"
1877   xml:lang="it-IT">
1878   <Msg ovf:msgid="info.os">Sistema operativo</Msg>
1879   ...
1880 </Strings>
```

1881 The embedded and external `Strings` elements may be interleaved, but they shall be placed at the end
 1882 of the `Envelope` element. If multiple occurrences of a `msgid` attribute with a given locale occur, a latter
 1883 value overwrites a former.

1884 10.3 Message Content in External File

1885 Starting with version 2.0 of this specification, the content of all localizable messages may be stored in an
 1886 external file using the optional `ovf:fileRef` attribute on the `Msg` element. For the `License` element on
 1887 `EulaSection` in particular, this allows inclusion of a standard license text file in unaltered form without
 1888 any XML header or footer.

1889 The `ovf:fileRef` attribute denotes the message content by identifying an existing `File` element in the
 1890 `References` element, the `File` element is identified by matching its `ovf:id` attribute value with the
 1891 `ovf:fileRef` attribute value. The content of an external file referenced using `ovf:fileRef` shall be
 1892 interpreted as plain text in UTF-8 Unicode.

1893 If the referenced file is not found, the embedded content of the `Msg` element shall be used.

1894 The optional `ovf:fileRef` attribute may appear on `Msg` elements in both internal and external `Strings`
 1895 resource bundles.

1896 EXAMPLE 5:

```
1897 <Envelope xml:lang="en-US">
1898   <References>
1899     <File ovf:id="license-en-US" ovf:href="license-en-US.txt"/>
1900     <File ovf:id="license-de-DE" ovf:href="license-de-DE.txt"/>
1901   </References>
1902   ...
1903 <VirtualSystem ovf:id="...">
```

```

1904     <EulaSection>
1905         <Info>Licensing agreement</Info>
1906         <License ovf:msgid="license">Unused</License>
1907     </EulaSection>
1908     ...
1909 </VirtualSystem>
1910 ...
1911 <Strings xml:lang="en-US">
1912     <Msg ovf:msgid="license" ovf:fileRef="license-en-US">Invalid license</Msg>
1913 </Strings>
1914 <Strings xml:lang="de-DE">
1915     <Msg ovf:msgid="license" ovf:fileRef="license-de-DE">Ihre Lizenz ist nicht
1916 gültig</Msg>
1917 </Strings>
1918 </Envelope>

```

1919 In the example above, the default license agreement is stored in plain text file `license-en-US.txt`,
 1920 while the license agreement for the `de-DE` locale is stored in file `license-de-DE.txt`.

1921 Note that the above mechanism works for all localizable elements and not just `License`.

1922 11 OVF Environment

1923 The OVF environment defines how the guest software and the deployment platform interact. This
 1924 environment allows the guest software to access information about the deployment platform, such as the
 1925 user-specified values for the properties defined in the OVF descriptor.

1926 The environment specification is split into a *protocol* part and a *transport* part. The *protocol* part defines
 1927 the format and semantics of an XML document that can be made accessible to the guest software. The
 1928 *transport* part defines how the information is communicated between the deployment platform and the
 1929 guest software.

1930 The `dsp8027_1.1.0.xsd` XML schema definition file for the OVF environment contains the elements
 1931 and attributes.

1932 11.1 Environment Document

1933 The environment document is an extensible XML document that is provided to the guest software about
 1934 the environment in which it is being executed. The way that the document is obtained depends on the
 1935 transport type.

1936 **EXAMPLE:** An example of the structure of the OVF environment document follows:

```

1937 <?xml version="1.0" encoding="UTF-8"?>
1938 <Environment xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
1939     xmlns:ovfenv="http://schemas.dmtf.org/ovf/environment/1"
1940     xmlns="http://schemas.dmtf.org/ovf/environment/1"
1941     ovfenv:id="identification of VM from OVF descriptor">
1942     <!-- Information about virtualization platform -->
1943     <PlatformSection>
1944         <Kind>Type of virtualization platform</Kind>
1945         <Version>Version of virtualization platform</Version>
1946         <Vendor>Vendor of virtualization platform</Vendor>
1947         <Locale>Language and country code</Locale>
1948         <TimeZone>Current timezone offset in minutes from UTC</TimeZone>
1949     </PlatformSection>
1950     <!-- Properties defined for this virtual machine -->
1951     <PropertySection>
1952         <Property ovfenv:key="key" ovfenv:value="value">
1953             <!-- More properties -->
1954     </PropertySection>

```

```

1955 <Entity ovfenv:id="id of sibling virtual system or virtual system collection">
1956   <PropertySection>
1957     <!-- Properties from sibling -->
1958   </PropertySection>
1959 </Entity>
1960 </Environment>
    
```

1961 The value of the `ovfenv:id` attribute of the `Environment` element shall match the value of the `ovf:id`
 1962 attribute of the `VirtualSystem` entity describing this virtual machine.

1963 The `PlatformSection` element contains optional information provided by the deployment platform.
 1964 Elements `Kind`, `Version`, and `Vendor` describe deployment platform vendor details; these elements are
 1965 experimental. Elements `Locale` and `TimeZone` describe the current locale and time zone; these
 1966 elements are experimental.

1967 The `PropertySection` element contains `Property` elements with key/value pairs corresponding to all
 1968 properties specified in the OVF descriptor for the current virtual machine, as well as properties specified
 1969 for the immediate parent `VirtualSystemCollection`, if one exists. The environment presents
 1970 properties as a simple list to make it easy for applications to parse. Furthermore, the single list format
 1971 supports the override semantics where a property on a `VirtualSystem` may override one defined on a
 1972 parent `VirtualSystemCollection`. The overridden property shall not be in the list. Overriding may
 1973 occur if a property in the current virtual machine and a property in the parent
 1974 `VirtualSystemCollection` has identical `ovf:key`, `ovf:class`, and `ovf:instance` attribute
 1975 values; see 9.5. In this case, the value of an overridden parent property may be obtained by adding a
 1976 differently named child property referencing the parent property with a macro; see 9.5.

1977 An `Entity` element shall exist for each sibling `VirtualSystem` and `VirtualSystemCollection`, if
 1978 any are present. The value of the `ovfenv:id` attribute of the `Entity` element shall match the value of
 1979 the `ovf:id` attribute of the sibling entity. The `Entity` elements contain the property key/value pairs in
 1980 the sibling's OVF environment documents, so the content of an `Entity` element for a particular sibling
 1981 shall contain the exact `PropertySection` seen by that sibling. This information can be used, for
 1982 example, to make configuration information such as IP addresses available to `VirtualSystems` being
 1983 part of a multi-tiered application.

1984 Table 8 shows the core sections that are defined.

1985 **Table 8 – Core Sections**

Section	Location	Multiplicity
<code>PlatformSection</code> Provides information from the deployment platform	Environment	Zero or one
<code>PropertySection</code> Contains key/value pairs corresponding to properties defined in the OVF descriptor	Environment Entity	Zero or one

1986 The environment document is extensible by providing new section types. A consumer of the document
 1987 should ignore unknown section types and elements.

1988 11.2 Transport

1989 The environment document information can be communicated in a number of ways to the guest software.
 1990 These ways are called transport types. The transport types are specified in the OVF descriptor by the
 1991 `ovf:transport` attribute of `VirtualHardwareSection`. Several transport types may be specified,
 1992 separated by a single space character, in which case an implementation is free to use any of them. The

- 1993 transport types define methods by which the environment document is communicated from the
1994 deployment platform to the guest software.
- 1995 To enable interoperability, this specification defines an "iso" transport type which all implementations
1996 that support CD-ROM devices are required to support. The iso transport communicates the environment
1997 document by making a dynamically generated ISO image available to the guest software. To support the
1998 iso transport type, prior to booting a virtual machine, an implementation shall make an ISO read-only
1999 disk image available as backing for a disconnected CD-ROM. If the iso transport is selected for a
2000 VirtualHardwareSection, at least one disconnected CD-ROM device shall be present in this section.
- 2001 The generated ISO image shall comply with the ISO 9660 specification with support for Joliet extensions.
- 2002 The ISO image shall contain the OVF environment for this particular virtual machine, and the environment
2003 shall be present in an XML file named ovf-env.xml that is contained in the root directory of the ISO
2004 image. The guest software can now access the information using standard guest operating system tools.
- 2005 If the virtual machine prior to booting had more than one disconnected CD-ROM, the guest software may
2006 have to scan connected CD-ROM devices in order to locate the ISO image containing the ovf-env.xml
2007 file.
- 2008 The ISO image containing the OVF environment shall be made available to the guest software on every
2009 boot of the virtual machine.
- 2010 Support for the "iso" transport type is not a requirement for virtual hardware architectures or guest
2011 operating systems which do not have CD-ROM device support.
- 2012 To be compliant with this specification, any transport format other than iso shall be given by a URI which
2013 identifies an unencumbered specification on how to use the transport. The specification need not be
2014 machine readable, but it shall be static and unique so that it may be used as a key by software reading an
2015 OVF descriptor to uniquely determine the format. The specification shall be sufficient for a skilled person
2016 to properly interpret the transport mechanism for implementing the protocols. The URIs should be
2017 resolvable.

ANNEX A (informative)

2018
2019
2020
2021

Symbols and Conventions

2022 XML examples use the XML namespace prefixes defined in Table 1. The XML examples use a style to
2023 not specify namespace prefixes on child elements. Note that XML rules define that child elements
2024 specified without namespace prefix are from the namespace of the parent element, and not from the
2025 default namespace of the XML document. Throughout the document, whitespace within XML element
2026 values is used for readability. In practice, a service can accept and strip leading and trailing whitespace
2027 within element values as if whitespace had not been used.

2028 Syntax definitions in this document use Augmented BNF (ABNF) as defined in IETF [RFC5234](#) with the
2029 following exceptions:

- 2030 • Rules separated by a bar (|) represent choices, instead of using a forward slash (/) as defined in
2031 ABNF.
- 2032 • Any characters must be processed case sensitively, instead of case-insensitively as defined in
2033 ABNF.
- 2034 • Whitespace (i.e., the space character U+0020 and the tab character U+0009) is allowed between
2035 syntactical elements, instead of assembling elements without whitespace as defined in ABNF.

2036

**ANNEX B
(normative)****OVF XSD**

2037
2038
2039
2040

2041 Normative copies of the XML schemas for this specification may be retrieved by resolving the following
2042 URLs:

2043
2044 <http://schemas.dmtf.org/ovf/envelope/2/dsp8023.xsd>
2045 <http://schemas.dmtf.org/ovf/environment/1/dsp8027.xsd>

2046 Any `xs:documentation` content in XML schemas for this specification is informative and provided only
2047 for convenience.

2048 Normative copies of the XML schemas for the WS-CIM mapping ([DSP0230](#)) of
2049 `CIM_ResourceAllocationSystemSettingsData`, `CIM_VirtualSystemSettingData`,
2050 `CIM_EthernetPortAllocationSettingData`, `CIM_StorageAllocationSettingData` and
2051 `CIM_OperatingSystem`, may be retrieved by resolving the following URLs:

2052
2053 http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/CIM_VirtualSystemSettingData.xsd
2054 [http://schemas.dmtf.org/wbem/wscim/1/cim-
2055 schema/2/CIM_ResourceAllocationSettingData.xsd](http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/CIM_ResourceAllocationSettingData.xsd)
2056 [http://schemas.dmtf.org/wbem/wscim/1/cim-
2057 schema/2/CIM_EthernetPortAllocationSettingData.xsd](http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/CIM_EthernetPortAllocationSettingData.xsd)
2058 http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/CIM_StorageAllocationSettingData.xsd

2059 This specification is based on the following CIM MOFs:

2060 `CIM_VirtualSystemSettingData.mof`
2061 `CIM_ResourceAllocationSettingData.mof`
2062 `CIM_EthernetPortAllocationSettingData.mof`
2063 `CIM_StorageAllocationSettingData.mof`
2064 `CIM_OperatingSystem.mof`
2065

ANNEX C (informative)

OVF Mime Type Registration Template

2070 Registration Template

2071 To: ietf-types@iana.org

2072 Subject: Registration of media type Application/OVF

2073 Type name: Application

2074 Subtype name: OVF

2075 Required parameters: none

2076 Optional parameters: none

2077 Encoding considerations: binary

2078 Security considerations:

- 2079 • An OVF package contains active content that is expected to be launched in a virtual machine.
2080 The OVF standard, section 5.1 states: “An OVF package may be signed by signing the manifest
2081 file. The digest of the manifest file is stored in a certificate file with extension .cert file along with
2082 the base64-encoded X.509 certificate. The .cert file shall have the same base name as the .ovf
2083 file and be a sibling of the .ovf file. A consumer of the OVF package shall verify the signature and
2084 should validate the certificate.
- 2085 • An OVF package may contain passwords as part of the configuration information. The OVF
2086 standard, section 9.5 states: “The optional Boolean attribute ovf:password indicates that the
2087 property value may contain sensitive information. The default value is FALSE. OVF
2088 implementations prompting for property values are advised to obscure these values when
2089 ovf:password is set to TRUE. This is similar to HTML text input of type password. Note that this
2090 mechanism affords limited security protection only. Although sensitive values are masked from
2091 casual observers, default values in the OVF descriptor and assigned values in the OVF
2092 environment are still passed in clear text. “

2093 Interoperability considerations:

- 2094 • OVF has demonstrated interoperability via multiple, interoperating implementations in the market.

2095 Published specification:

- 2096 • DSP0243_2.0.0.pdf

2097 Applications that use this media type:

- 2098 • Implementations of the DMTF Standard: Cloud Infrastructure Management Interface (CIMI)
2099 (DSP0263_1.0.0.pdf)
- 2100 • Implementations of the SNIA Cloud Data Management Interface (CDMI) – OVF Extension

2101 Additional information:

- 2102 • Magic number(s): none
- 2103 • File extension(s): .ova
- 2104 • Macintosh file type code(s): none
- 2105 • Person & email address to contact for further information:

- 2106 • Intended usage: (One of COMMON, LIMITED USE or OBSOLETE.)
- 2107 • Restrictions on usage: (Any restrictions on where the media type can be used go here.)
- 2108 • Author:
- 2109 • Change controller:

ANNEX D (informative)

Network Port Profile Examples

D.1 Example 1 (OVF Descriptor for One Virtual System and One Network with an Inlined Network Port Profile)

The example below shows an OVF descriptor that describes a virtual system and a network it connects to. The virtual system description in this example uses an inlined network port profile that is described as an XML element that contains child XML elements from epasd namespace. The network described in the network section uses the same network port profile description. The network port profile described in this example is used to reserve 1 Gbps of bandwidth.

```

2114 <?xml version="1.0" encoding="UTF-8"?>
2115
2116 <Envelope xsi:schemaLocation="http://schemas.dmtf.org/ovf/envelope/2
2117 file:///C:/dsp8023_2.0.0_wgv0.9.5.xsd" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
2118 xmlns:ovf="http://schemas.dmtf.org/ovf/envelope/2" xmlns="http://schemas.dmtf.org/ovf/envelope/2"
2119 xmlns:vssd="http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/CIM_VirtualSystemSettingData"
2120 xmlns:rasd="http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/CIM_ResourceAllocationSettingData"
2121 xmlns:epasd="http://schemas.dmtf.org/wbem/wscim/1/cim-
2122 schema/2/CIM_EthernetPortAllocationSettingData"
2123 xmlns:sasd="http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/CIM_StorageAllocationSettingData">
2124
2125 <!-- References to all external files -->
2126
2127 <References>
2128
2129 <File ovf:id="file1" ovf:href="vmdisk1.vmdk" ovf:size="2000000000"/>
2130
2131 </References>
2132
2133 <!-- Describes meta-information for all virtual disks in the package -->
2134
2135 <DiskSection>
2136
2137 <Info>Describes the set of virtual disks</Info>
2138
2139 <Disk ovf:diskId="vmdisk1" ovf:fileRef="file1" ovf:capacity="4294967296"
2140 ovf:format="http://www.examplecompany.com/interfaces/specifications/vmdk.html#sparse"/>
2141
2142 </DiskSection>
2143
2144 <!-- Describes all networks used in the package -->
2145
2146 <NetworkSection>
2147
2148 <Info>List of logical networks used in the package</Info>
2149
2150 <Network ovf:name="VM Network">
2151
2152 <Description>The network that the VMs connect to</Description>
2153
2154 <NetworkPortProfile>
2155
2156 <!-- Network port profile describing bandwidth reservation. Network port profile
2157 is identified by UUID. -->
2158
2159 <Item>
2160
2161 <epasd:AllocationUnits>bit / second * 10^9</epasd:AllocationUnits>
2162
2163 <epasd:ElementName>Network Port Profile 1</epasd:ElementName>

```

```

2151         <epasd:InstanceID>1</epasd:InstanceID>
2152         <epasd:NetworkPortProfileID>aaaaaaaa-bbbb-cccc-dddd-
2153 eeeeeeeeeee</epasd:NetworkPortProfileID>
2154         <epasd:NetworkPortProfileIDType>3</epasd:NetworkPortProfileIDType>
2155         <epasd:Reservation>1</epasd:Reservation>
2156     </Item>
2157 </NetworkPortProfile>
2158 </Network>
2159 </NetworkSection>
2160 <VirtualSystem ovf:id="vm">
2161     <Info>Describes a virtual machine</Info>
2162     <Name>Virtual Appliance One</Name>
2163     <ProductSection>
2164         <Info>Describes product information for the appliance</Info>
2165         <Product>The Great Appliance</Product>
2166         <Vendor>Some Great Corporation</Vendor>
2167         <Version>13.00</Version>
2168         <FullVersion>13.00-b5</FullVersion>
2169         <ProductUrl>http://www.somegreatcorporation.com/greatappliance</ProductUrl>
2170         <VendorUrl>http://www.somegreatcorporation.com/</VendorUrl>
2171         <Property ovf:key="admin.email" ovf:type="string">
2172             <Description>Email address of administrator</Description>
2173         </Property>
2174         <Property ovf:key="app.ip" ovf:type="string" ovf:defaultValue="192.168.0.10">
2175             <Description>The IP address of this appliance</Description>
2176         </Property>
2177     </ProductSection>
2178     <AnnotationSection ovf:required="false">
2179         <Info>A random annotation on this service. It can be ignored</Info>
2180         <Annotation>Contact customer support if you have any problems</Annotation>
2181     </AnnotationSection>
2182     <EulaSection>
2183         <Info>License information for the appliance</Info>
2184         <License>Insert your favorite license here</License>
2185     </EulaSection>
2186     <VirtualHardwareSection>

```

```

2187     <Info>Memory = 4 GB, CPU = 1 GHz, Disk = 100 GB, 1 Ethernet nic</Info>
2188     <Item>
2189         <rasd:AllocationUnits>Hertz*10^9</rasd:AllocationUnits>
2190         <rasd:Description>Virtual CPU</rasd:Description>
2191         <rasd:ElementName>1 GHz virtual CPU</rasd:ElementName>
2192         <rasd:InstanceID>1</rasd:InstanceID>
2193         <rasd:Reservation>1</rasd:Reservation>
2194         <rasd:ResourceType>3</rasd:ResourceType>
2195         <rasd:VirtualQuantity>1</rasd:VirtualQuantity>
2196     </Item>
2197     <Item>
2198         <rasd:AllocationUnits>byte*2^30</rasd:AllocationUnits>
2199         <rasd:Description>Memory</rasd:Description>
2200         <rasd:ElementName>1 GByte of memory</rasd:ElementName>
2201         <rasd:InstanceID>2</rasd:InstanceID>
2202         <rasd:ResourceType>4</rasd:ResourceType>
2203         <rasd:VirtualQuantity>1</rasd:VirtualQuantity>
2204     </Item>
2205     <EthernetPortItem>
2206         <epasd:Address>00-16-8B-DB-00-5E</epasd:Address>
2207         <epasd:AllocationUnits>bit / second * 10^9 </epasd:AllocationUnits>
2208         <epasd:Connection>VM Network</epasd:Connection>
2209         <epasd:Description>Virtual NIC</epasd:Description>
2210         <epasd:ElementName>Ethernet Port</epasd:ElementName>
2211
2212         <epasd:InstanceID>3</epasd:InstanceID>
2213         <epasd:NetworkPortProfileID>aaaaaaaa-bbbb-cccc-dddd-
2214 eeeeeeeeeee</epasd:NetworkPortProfileID>
2215         <epasd:NetworkPortProfileIDType>3</epasd:NetworkPortProfileIDType>
2216         <epasd:Reservation>1</epasd:Reservation>
2217         <epasd:ResourceType>10</epasd:ResourceType>
2218         <epasd:VirtualQuantityUnits>1</epasd:VirtualQuantityUnits>
2219     </EthernetPortItem>
2220     <StorageItem>
2221         <sasd:AllocationUnits>byte*2^30</sasd:AllocationUnits>
2222         <sasd:Description>Virtual Disk</sasd:Description>

```

```

2223         <sasd:ElementName>100 GByte Virtual Disk</sasd:ElementName>
2224         <sasd:InstanceID>4</sasd:InstanceID>
2225         <sasd:Reservation>100</sasd:Reservation>
2226         <sasd:ResourceType>31</sasd:ResourceType>
2227         <sasd:VirtualQuantity>1</sasd:VirtualQuantity>
2228     </StorageItem>
2229 </VirtualHardwareSection>
2230 <OperatingSystemSection ovf:id="58" ovf:required="false">
2231     <Info>Guest Operating System</Info>
2232     <Description>OS</Description>
2233 </OperatingSystemSection>
2234 </VirtualSystem>
2235 </Envelope>

```

2236 D.2 Example 2 (OVF Descriptor for One Virtual System and One Network with a 2237 Locally Referenced Network Port Profile)

2238 The example below shows an OVF descriptor that describes a virtual system and a network it connects
2239 to. The virtual system description in this example uses a network port profile that is described in a local
2240 file that is contained in the same OVF package. The network described in the network section uses the
2241 same network port profile description. The network port profile described in this example is used to
2242 reserve 1 Gbps of bandwidth.

```

2243 <?xml version="1.0" encoding="UTF-8"?>
2244 <Envelope xsi:schemaLocation="http://schemas.dmtf.org/ovf/envelope/2
2245 file:///C:/dsp8023_2.0.0_wgv0.9.5.xsd" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
2246 xmlns:ovf="http://schemas.dmtf.org/ovf/envelope/2" xmlns="http://schemas.dmtf.org/ovf/envelope/2"
2247 xmlns:vssd="http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/CIM_VirtualSystemSettingData"
2248 xmlns:rasd="http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/CIM_ResourceAllocationSettingData"
2249 xmlns:epasd="http://schemas.dmtf.org/wbem/wscim/1/cim-
2250 schema/2/CIM_EthernetPortAllocationSettingData"
2251 xmlns:sasd="http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/CIM_StorageAllocationSettingData">
2252 <!-- References to all external files -->
2253 <References>
2254     <File ovf:id="file1" ovf:href="vmdisk1.vmdk" ovf:size="2000000000"/>
2255     <File ovf:id="networkportprofile1" ovf:href="NetworkPortProfile1.xml"/>
2256 </References>
2257 <!-- Describes meta-information for all virtual disks in the package -->
2258 <DiskSection>
2259     <Info>Describes the set of virtual disks</Info>
2260     <Disk ovf:diskId="vmdisk1" ovf:fileRef="file1" ovf:capacity="4294967296"
2261 ovf:format="http://www.examplecompany.com/interfaces/specifications/vmdk.html#sparse"/>
2262 </DiskSection>
2263 <!-- Describes all networks used in the package -->

```

```

2264 <NetworkSection>
2265     <Info>List of logical networks used in the package</Info>
2266     <Network ovf:name="VM Network">
2267         <Description>The network that VMs connect to</Description>
2268         <NetworkPortProfileURI>file:networkportprofile1</NetworkPortProfileURI>
2269     </Network>
2270 </NetworkSection>
2271 <VirtualSystem ovf:id="vm">
2272     <Info>Describes a virtual machine</Info>
2273     <Name>Virtual Appliance One</Name>
2274     <ProductSection>
2275         <Info>Describes product information for the appliance</Info>
2276         <Product>The Great Appliance</Product>
2277         <Vendor>Some Great Corporation</Vendor>
2278         <Version>13.00</Version>
2279         <FullVersion>13.00-b5</FullVersion>
2280         <ProductUrl>http://www.somegreatcorporation.com/greatappliance</ProductUrl>
2281         <VendorUrl>http://www.somegreatcorporation.com/</VendorUrl>
2282         <Property ovf:key="admin.email" ovf:type="string">
2283             <Description>Email address of administrator</Description>
2284         </Property>
2285         <Property ovf:key="app.ip" ovf:type="string" ovf:defaultValue="192.168.0.10">
2286             <Description>The IP address of this appliance</Description>
2287         </Property>
2288     </ProductSection>
2289     <AnnotationSection ovf:required="false">
2290         <Info>A random annotation on this service. It can be ignored</Info>
2291         <Annotation>Contact customer support if you have any problems</Annotation>
2292     </AnnotationSection>
2293     <EulaSection>
2294         <Info>License information for the appliance</Info>
2295         <License>Insert your favorite license here</License>
2296     </EulaSection>
2297     <VirtualHardwareSection>
2298         <Info>Memory = 4 GB, CPU = 1 GHz, Disk = 100 GB, 1 Ethernet nic</Info>
2299         <Item>

```

```

2300         <rasd:AllocationUnits>Hertz*10^9</rasd:AllocationUnits>
2301         <rasd:Description>Virtual CPU</rasd:Description>
2302         <rasd:ElementName>1 GHz virtual CPU</rasd:ElementName>
2303         <rasd:InstanceID>1</rasd:InstanceID>
2304         <rasd:Reservation>1</rasd:Reservation>
2305         <rasd:ResourceType>3</rasd:ResourceType>
2306         <rasd:VirtualQuantity>1</rasd:VirtualQuantity>
2307     </Item>
2308     <Item>
2309         <rasd:AllocationUnits>byte*2^30</rasd:AllocationUnits>
2310         <rasd:Description>Memory</rasd:Description>
2311         <rasd:ElementName>1 GByte of memory</rasd:ElementName>
2312         <rasd:InstanceID>2</rasd:InstanceID>
2313         <rasd:ResourceType>4</rasd:ResourceType>
2314         <rasd:VirtualQuantity>1</rasd:VirtualQuantity>
2315     </Item>
2316     <EthernetPortItem>
2317         <epasd:Address>00-16-8B-DB-00-5E</epasd:Address>
2318         <epasd:Connection>VM Network</epasd:Connection>
2319         <epasd:Description>Virtual NIC</epasd:Description>
2320         <epasd:ElementName>Ethernet Port</epasd:ElementName>
2321
2322         <epasd:InstanceID>3</epasd:InstanceID>
2323         <epasd:NetworkPortProfileID>file:networkportprofile1</epasd:NetworkPortProfileID>
2324         <epasd:NetworkPortProfileIDType>2</epasd:NetworkPortProfileIDType>
2325         <epasd:ResourceType>10</epasd:ResourceType>
2326         <epasd:VirtualQuantityUnits>1</epasd:VirtualQuantityUnits>
2327     </EthernetPortItem>
2328     <StorageItem>
2329         <sasd:AllocationUnits>byte*2^30</sasd:AllocationUnits>
2330         <sasd:Description>Virtual Disk</sasd:Description>
2331         <sasd:ElementName>100 GByte Virtual Disk</sasd:ElementName>
2332         <sasd:InstanceID>4</sasd:InstanceID>
2333         <sasd:Reservation>100</sasd:Reservation>
2334         <sasd:ResourceType>31</sasd:ResourceType>
2335         <sasd:VirtualQuantity>1</sasd:VirtualQuantity>

```

```

2336         </StorageItem>
2337     </VirtualHardwareSection>
2338     <OperatingSystemSection ovf:id="58" ovf:required="false">
2339         <Info>Guest Operating System</Info>
2340         <Description>OS</Description>
2341     </OperatingSystemSection>
2342 </VirtualSystem>
2343 </Envelope>

```

2344 **D.3 Example 3 (OVF Descriptor for One Virtual System and One Network with a** 2345 **Network Port Profile referenced by a URI)**

2346 The example below shows an OVF descriptor that describes a virtual system and a network it connects
2347 to. The virtual system description in this example uses a network port profile that is described by a URI.
2348 The network described in the network section uses the same network port profile description. The
2349 network port profile described in this example is used to reserve 1 Gbps of bandwidth.

```

2350 <?xml version="1.0" encoding="UTF-8"?>
2351 <Envelope xsi:schemaLocation="http://schemas.dmtf.org/ovf/envelope/2
2352 file:///C:/dsp8023_2.0.0_wgv0.9.5.xsd" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
2353 xmlns:ovf="http://schemas.dmtf.org/ovf/envelope/2" xmlns="http://schemas.dmtf.org/ovf/envelope/2"
2354 xmlns:vssd="http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/CIM_VirtualSystemSettingData"
2355 xmlns:rasd="http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/CIM_ResourceAllocationSettingData"
2356 xmlns:epasd="http://schemas.dmtf.org/wbem/wscim/1/cim-
2357 schema/2/CIM_EthernetPortAllocationSettingData"
2358 xmlns:sasd="http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/CIM_StorageAllocationSettingData">
2359 <!-- References to all external files -->
2360     <References>
2361         <File ovf:id="file1" ovf:href="vmdisk1.vmdk" ovf:size="2000000000"/>
2362     </References>
2363     <!-- Describes meta-information for all virtual disks in the package -->
2364     <DiskSection>
2365         <Info>Describes the set of virtual disks</Info>
2366         <Disk ovf:diskId="vmdisk1" ovf:fileRef="file1" ovf:capacity="4294967296"
2367 ovf:format="http://www.examplecompany.com/interfaces/specifications/vmdk.html#sparse"/>
2368     </DiskSection>
2369     <!-- Describes all networks used in the package -->
2370     <NetworkSection>
2371         <Info>List of logical networks used in the package</Info>
2372         <Network ovf:name="VM Network">
2373             <Description>The network that the VMs connect to</Description>
2374             <NetworkPortProfileURI>http://www.dmtf.org/networkportprofiles/networkportprofile1.xml</Netwo
2375 rkPortProfileURI>
2376         </Network>
2377     </NetworkSection>

```

```
2378 </NetworkSection>
2379 <VirtualSystem ovf:id="vm">
2380   <Info>Describes a virtual machine</Info>
2381   <Name>Virtual Appliance One</Name>
2382   <ProductSection>
2383     <Info>Describes product information for the appliance</Info>
2384     <Product>The Great Appliance</Product>
2385     <Vendor>Some Great Corporation</Vendor>
2386     <Version>13.00</Version>
2387     <FullVersion>13.00-b5</FullVersion>
2388     <ProductUrl>http://www.somegreatcorporation.com/greatappliance</ProductUrl>
2389     <VendorUrl>http://www.somegreatcorporation.com/</VendorUrl>
2390     <Property ovf:key="admin.email" ovf:type="string">
2391       <Description>Email address of administrator</Description>
2392     </Property>
2393     <Property ovf:key="app.ip" ovf:type="string" ovf:defaultValue="192.168.0.10">
2394       <Description>The IP address of this appliance</Description>
2395     </Property>
2396   </ProductSection>
2397   <AnnotationSection ovf:required="false">
2398     <Info>A random annotation on this service. It can be ignored</Info>
2399     <Annotation>Contact customer support if you have any problems</Annotation>
2400   </AnnotationSection>
2401   <EulaSection>
2402     <Info>License information for the appliance</Info>
2403     <License>Insert your favorite license here</License>
2404   </EulaSection>
2405   <VirtualHardwareSection>
2406     <Info>Memory = 4 GB, CPU = 1 GHz, Disk = 100 GB, 1 Ethernet nic</Info>
2407     <Item>
2408       <rasd:AllocationUnits>Hertz*10^9</rasd:AllocationUnits>
2409       <rasd:Description>Virtual CPU</rasd:Description>
2410       <rasd:ElementName>1 GHz virtual CPU</rasd:ElementName>
2411       <rasd:InstanceID>1</rasd:InstanceID>
2412       <rasd:Reservation>1</rasd:Reservation>
2413       <rasd:ResourceType>3</rasd:ResourceType>
```



```

2414         <rasd:VirtualQuantity>1</rasd:VirtualQuantity>
2415     </Item>
2416     <Item>
2417         <rasd:AllocationUnits>byte*2^30</rasd:AllocationUnits>
2418         <rasd:Description>Memory</rasd:Description>
2419         <rasd:ElementName>1 GByte of memory</rasd:ElementName>
2420         <rasd:InstanceID>2</rasd:InstanceID>
2421         <rasd:ResourceType>4</rasd:ResourceType>
2422         <rasd:VirtualQuantity>1</rasd:VirtualQuantity>
2423     </Item>
2424     <EthernetPortItem>
2425         <epasd:Address>00-16-8B-DB-00-5E</epasd:Address>
2426         <epasd:Connection>VM Network</epasd:Connection>
2427         <epasd:Description>Virtual NIC</epasd:Description>
2428         <epasd:ElementName>Ethernet Port</epasd:ElementName>
2429
2430         <epasd:InstanceID>3</epasd:InstanceID>
2431
2432         <epasd:NetworkPortProfileID>http://www.dmtf.org/networkportprofiles/networkportprofile1.xml</
2433     epasd:NetworkPortProfileID>
2434         <epasd:NetworkPortProfileIDType>2</epasd:NetworkPortProfileIDType>
2435         <epasd:ResourceType>10</epasd:ResourceType>
2436         <epasd:VirtualQuantityUnits>1</epasd:VirtualQuantityUnits>
2437     </EthernetPortItem>
2438     <StorageItem>
2439         <sasd:AllocationUnits>byte*2^30</sasd:AllocationUnits>
2440         <sasd:Description>Virtual Disk</sasd:Description>
2441         <sasd:ElementName>100 GByte Virtual Disk</sasd:ElementName>
2442         <sasd:InstanceID>4</sasd:InstanceID>
2443         <sasd:Reservation>100</sasd:Reservation>
2444         <sasd:ResourceType>31</sasd:ResourceType>
2445         <sasd:VirtualQuantity>1</sasd:VirtualQuantity>
2446     </StorageItem>
2447 </VirtualHardwareSection>
2448 <OperatingSystemSection ovf:id="58" ovf:required="false">
2449     <Info>Guest Operating System</Info>
2450     <Description>OS</Description>

```

```

2451         </OperatingSystemSection>
2452     </VirtualSystem>
2453 </Envelope>

```

2454 **D.4 Example 4 (OVF Descriptor for Two Virtual Systems and One Network with** 2455 **Two Network Port Profiles referenced by URIs)**

2456 The example below shows an OVF descriptor that describes two virtual systems and a network they
2457 connect to. Each virtual system description in this example uses a network port profile that is described
2458 by a URI. The network described in the network section uses the same two network port profiles. The two
2459 network port profiles described in this example are used to reserve 1 Gbps of bandwidth and describe
2460 general network traffic respectively. Annex D.5 and A.1 are examples of these network port profiles.

```

2461 <?xml version="1.0" encoding="UTF-8"?>
2462 <Envelope xsi:schemaLocation="http://schemas.dmtf.org/ovf/envelope/2
2463 file:///C:/dsp8023_2.0.0_wgv0.9.5.xsd" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
2464 xmlns:ovf="http://schemas.dmtf.org/ovf/envelope/2" xmlns="http://schemas.dmtf.org/ovf/envelope/2"
2465 xmlns:vssd="http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/CIM_VirtualSystemSettingData"
2466 xmlns:rasd="http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/CIM_ResourceAllocationSettingData"
2467 xmlns:epasd="http://schemas.dmtf.org/wbem/wscim/1/cim-
2468 schema/2/CIM_EthernetPortAllocationSettingData"
2469 xmlns:sasd="http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/CIM_StorageAllocationSettingData">
2470 <!-- References to all external files -->
2471     <References>
2472         <File ovf:id="file1" ovf:href="vmdisk1.vmdk" ovf:size="2000000000"/>
2473     </References>
2474     <!-- Describes meta-information for all virtual disks in the package -->
2475     <DiskSection>
2476         <Info>Describes the set of virtual disks</Info>
2477         <Disk ovf:diskId="vmdisk1" ovf:fileRef="file1" ovf:capacity="4294967296"
2478 ovf:format="http://www.examplecompany.com/interfaces/specifications/vmdk.html#sparse"/>
2479     </DiskSection>
2480     <!-- Describes all networks used in the package -->
2481     <NetworkSection>
2482         <Info>List of logical networks used in the package</Info>
2483         <Network ovf:name="VM Network">
2484             <Description>The network that the VMs connect to</Description>
2485             <!-- Network port profile for storage traffic -->
2486             <NetworkPortProfileURI>http://www.dmtf.org/networkportprofiles/networkportprofile1.xml</Netwo
2487 rkPortProfileURI>
2488             <!-- Network port profile for networking traffic -->
2489             <NetworkPortProfileURI>http://www.dmtf.org/networkportprofiles/networkportprofile2.xml</Netwo
2490 rkPortProfileURI>
2491         </Network>
2492     </NetworkSection>
2493 </Envelope>

```

```

2494 </NetworkSection>
2495 <VirtualSystemCollection ovf:id="vsc1">
2496   <Info>Collection of 2 VMs</Info>
2497   <VirtualSystem ovf:id="storage server">
2498     <Info>Describes a virtual machine</Info>
2499     <Name>Virtual Appliance One</Name>
2500     <ProductSection>
2501       <Info>Describes product information for the appliance</Info>
2502       <Product>The Great Appliance</Product>
2503       <Vendor>Some Great Corporation</Vendor>
2504       <Version>13.00</Version>
2505       <FullVersion>13.00-b5</FullVersion>
2506       <ProductUrl>http://www.somegreatcorporation.com/greatappliance</ProductUrl>
2507       <VendorUrl>http://www.somegreatcorporation.com/</VendorUrl>
2508       <Property ovf:key="admin.email" ovf:type="string">
2509         <Description>Email address of administrator</Description>
2510       </Property>
2511       <Property ovf:key="app.ip" ovf:type="string" ovf:defaultValue="192.168.0.10">
2512         <Description>The IP address of this appliance</Description>
2513       </Property>
2514     </ProductSection>
2515     <AnnotationSection ovf:required="false">
2516       <Info>A random annotation on this service. It can be ignored</Info>
2517       <Annotation>Contact customer support if you have any problems</Annotation>
2518     </AnnotationSection>
2519     <EulaSection>
2520       <Info>License information for the appliance</Info>
2521       <License>Insert your favorite license here</License>
2522     </EulaSection>
2523     <VirtualHardwareSection>
2524       <Info>Memory = 4 GB, CPU = 1 GHz, Disk = 100 GB, 1 Ethernet nic</Info>
2525       <Item>
2526         <rasd:AllocationUnits>Hertz*10^9</rasd:AllocationUnits>
2527         <rasd:Description>Virtual CPU</rasd:Description>
2528         <rasd:ElementName>1 GHz virtual CPU</rasd:ElementName>
2529         <rasd:InstanceID>1</rasd:InstanceID>

```

```

2530         <rasd:Reservation>1</rasd:Reservation>
2531         <rasd:ResourceType>3</rasd:ResourceType>
2532         <rasd:VirtualQuantity>1</rasd:VirtualQuantity>
2533     </Item>
2534     <Item>
2535         <rasd:AllocationUnits>byte*2^30</rasd:AllocationUnits>
2536         <rasd:Description>Memory</rasd:Description>
2537         <rasd:ElementName>1 GByte of memory</rasd:ElementName>
2538         <rasd:InstanceID>2</rasd:InstanceID>
2539         <rasd:ResourceType>4</rasd:ResourceType>
2540         <rasd:VirtualQuantity>1</rasd:VirtualQuantity>
2541     </Item>
2542     <EthernetPortItem>
2543         <epasd:Address>00-16-8B-DB-00-5E</epasd:Address>
2544         <epasd:Connection>VM Network</epasd:Connection>
2545         <epasd:Description>Virtual NIC</epasd:Description>
2546         <epasd:ElementName>Ethernet Port</epasd:ElementName>
2547         <epasd:InstanceID>3</epasd:InstanceID>
2550         <epasd:NetworkPortProfileID>http://www.dmtf.org/networkportprofiles/networkportprofile1.xml</
2551         epasd:NetworkPortProfileID>
2552         <epasd:NetworkPortProfileIDType>2</epasd:NetworkPortProfileIDType>
2553         <epasd:ResourceType>10</epasd:ResourceType>
2554         <epasd:VirtualQuantityUnits>1</epasd:VirtualQuantityUnits>
2555     </EthernetPortItem>
2556     <StorageItem>
2557         <sasd:AllocationUnits>byte*2^30</sasd:AllocationUnits>
2558         <sasd:Description>Virtual Disk</sasd:Description>
2559         <sasd:ElementName>100 GByte Virtual Disk</sasd:ElementName>
2560         <sasd:InstanceID>4</sasd:InstanceID>
2561         <sasd:Reservation>100</sasd:Reservation>
2562         <sasd:ResourceType>31</sasd:ResourceType>
2563         <sasd:VirtualQuantity>1</sasd:VirtualQuantity>
2564     </StorageItem>
2565 </VirtualHardwareSection>
2566

```

```
2567     <OperatingSystemSection ovf:id="58" ovf:required="false">
2568         <Info>Guest Operating System</Info>
2569         <Description>OS</Description>
2570     </OperatingSystemSection>
2571 </VirtualSystem>
2572 <VirtualSystem ovf:id="web-server">
2573     <Info>Describes a virtual machine</Info>
2574     <Name>Virtual Appliance Two</Name>
2575     <ProductSection>
2576         <Info>Describes product information for the appliance</Info>
2577         <Product>The Great Appliance</Product>
2578         <Vendor>Some Great Corporation</Vendor>
2579         <Version>13.00</Version>
2580         <FullVersion>13.00-b5</FullVersion>
2581         <ProductUrl>http://www.somegreatcorporation.com/greatappliance</ProductUrl>
2582         <VendorUrl>http://www.somegreatcorporation.com/</VendorUrl>
2583         <Property ovf:key="admin.email" ovf:type="string">
2584             <Description>Email address of administrator</Description>
2585         </Property>
2586         <Property ovf:key="app.ip" ovf:type="string" ovf:defaultValue="192.168.0.10">
2587             <Description>The IP address of this appliance</Description>
2588         </Property>
2589     </ProductSection>
2590     <AnnotationSection ovf:required="false">
2591         <Info>A random annotation on this service. It can be ignored</Info>
2592         <Annotation>Contact customer support if you have any problems</Annotation>
2593     </AnnotationSection>
2594     <EulaSection>
2595         <Info>License information for the appliance</Info>
2596         <License>Insert your favorite license here</License>
2597     </EulaSection>
2598     <VirtualHardwareSection>
2599         <Info>Memory = 4 GB, CPU = 1 GHz, Disk = 100 GB, 1 Ethernet nic</Info>
2600         <Item>
2601             <rasd:AllocationUnits>Hertz*10^9</rasd:AllocationUnits>
2602             <rasd:Description>Virtual CPU</rasd:Description>
```

```

2603         <rasd:ElementName>1 GHz virtual CPU</rasd:ElementName>
2604         <rasd:InstanceID>1</rasd:InstanceID>
2605         <rasd:Reservation>1</rasd:Reservation>
2606         <rasd:ResourceType>3</rasd:ResourceType>
2607         <rasd:VirtualQuantity>1</rasd:VirtualQuantity>
2608     </Item>
2609     <Item>
2610         <rasd:AllocationUnits>byte*2^30</rasd:AllocationUnits>
2611         <rasd:Description>Memory</rasd:Description>
2612         <rasd:ElementName>1 GByte of memory</rasd:ElementName>
2613         <rasd:InstanceID>2</rasd:InstanceID>
2614         <rasd:ResourceType>4</rasd:ResourceType>
2615         <rasd:VirtualQuantity>1</rasd:VirtualQuantity>
2616     </Item>
2617     <EthernetPortItem>
2618         <epasd:Address>00-16-8B-DB-00-5F</epasd:Address>
2619         <epasd:Connection>VM Network</epasd:Connection>
2620         <epasd:Description>Virtual NIC</epasd:Description>
2621
2622         <epasd:ElementName>Ethernet Port</epasd:ElementName>
2623         <!-- Virtual NIC for networking traffic -->
2624         <epasd:InstanceID>3</epasd:InstanceID>
2625
2626         <epasd:NetworkPortProfileID>http://www.dmtf.org/networkportprofiles/networkportprofile2.xml</
2627     epasd:NetworkPortProfileID>
2628         <epasd:NetworkPortProfileIDType>2</epasd:NetworkPortProfileIDType>
2629         <epasd:ResourceType>10</epasd:ResourceType>
2630         <epasd:VirtualQuantityUnits>1</epasd:VirtualQuantityUnits>
2631     </EthernetPortItem>
2632     <StorageItem>
2633         <sasd:AllocationUnits>byte*2^30</sasd:AllocationUnits>
2634         <sasd:Description>Virtual Disk</sasd:Description>
2635         <sasd:ElementName>100 GByte Virtual Disk</sasd:ElementName>
2636         <sasd:InstanceID>4</sasd:InstanceID>
2637         <sasd:Reservation>100</sasd:Reservation>
2638         <sasd:ResourceType>31</sasd:ResourceType>
2639         <sasd:VirtualQuantity>1</sasd:VirtualQuantity>

```

```

2640         </StorageItem>
2641     </VirtualHardwareSection>
2642     <OperatingSystemSection ovf:id="58" ovf:required="false">
2643         <Info>Guest Operating System</Info>
2644         <Description>OS</Description>
2645     </OperatingSystemSection>
2646 </VirtualSystem>
2647 </VirtualSystemCollection>
2648 </Envelope>

```

2649 D.5 Example 5 (networkportprofile1.xml)

2650

2651 Network Port profile example for bandwidth reservation.

```

2652 <?xml version="1.0" encoding="UTF-8"?>
2653 <NetworkPortProfile xsi:schemaLocation="http://schemas.dmtf.org/ovf/networkportprofile/1
2654 http://schemas.dmtf.org/ovf/networkportprofile/1/dsp8049.xsd"
2655 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
2656 xmlns="http://schemas.dmtf.org/ovf/networkportprofile/1"
2657 xmlns:rasd="http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/CIM_ResourceAllocationSettingData"
2658 xmlns:epasd="http://schemas.dmtf.org/wbem/wscim/1/cim-
2659 schema/2/CIM_EthernetPortAllocationSettingData">
2660     <Item>
2661         <epasd:AllocationUnits>bit / second * 10^9</epasd:AllocationUnits>
2662         <epasd:ElementName>Network Port Profile 1</epasd:ElementName>
2663         <epasd:InstanceID>1</epasd:InstanceID>
2664         <epasd:NetworkPortProfileID>aaaaaaaa-bbbb-cccc-dddd-
2665 eeeeeeeeeee</epasd:NetworkPortProfileID>
2666         <epasd:NetworkPortProfileIDType>3</epasd:NetworkPortProfileIDType>
2667         <epasd:Reservation>1</epasd:Reservation>
2668     </Item>
2669 </NetworkPortProfile>

```

2670 D.6 Example 6 (networkportprofile2.xml)

2671 Network Port Profile example showing priority setting.

```

2672 <?xml version="1.0" encoding="UTF-8"?>
2673 <NetworkPortProfile xsi:schemaLocation="http://schemas.dmtf.org/ovf/networkportprofile/1
2674 http://schemas.dmtf.org/ovf/networkportprofile/1/dsp8049.xsd"
2675 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
2676 xmlns="http://schemas.dmtf.org/ovf/networkportprofile/1"
2677 xmlns:rasd="http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/CIM_ResourceAllocationSettingData"
2678 xmlns:epasd="http://schemas.dmtf.org/wbem/wscim/1/cim-
2679 schema/2/CIM_EthernetPortAllocationSettingData">

```

```
2680     <Item>
2681         <epasd:AllowedPriorities>0</epasd:AllowedPriorities>
2682         <epasd:AllowedPriorities>1</epasd:AllowedPriorities>
2683         <epasd:DefaultPriority>0</epasd:DefaultPriority>
2684         <epasd:ElementName>Network Port Profile 2</epasd:ElementName>
2685         <epasd:InstanceID>2</epasd:InstanceID>
2686         <epasd:NetworkPortProfileID>aaaaaaaa-bbbb-cccc-dddd-
2687 ffffffff</epasd:NetworkPortProfileID>
2688         <epasd:NetworkPortProfileIDType>3</epasd:NetworkPortProfileIDType>
2689     </Item>
2690 </NetworkPortProfile>
2691
```


**ANNEX E
(informative)**

Change Log

2692
2693
2694
2695

Version	Date	Description
1.0.0	2009-02-22	
1.1.0	2010-01-12	DMTF Standard release
2.0.0	2012-12-13	DMTF Standard release
2.0.1	2013-08-22	DMTF Standard – subclause 9.10 initialBootStopDelay stated meaning of zero value – subclause 9.4 Addressed use of ':' and '.' characters

2696

Bibliography

2697

2698 ISO 9660, *Joliet Extensions Specification*, May 1995,
2699 <http://littlesvr.ca/isomaster/resources/JolietSpecification.html>

2700 W3C, *Best Practices for XML Internationalization*, October 2008,
2701 <http://www.w3.org/TR/2008/NOTE-xml-i18n-bp-20080213/>

2702 DMTF DSP1044, *Processor Device Resource Virtualization Profile 1.0*
2703 http://www.dmtf.org/standards/published_documents/DSP1044_1.0.pdf

2704 DMTF DSP1045, *Memory Resource Virtualization Profile 1.0*
2705 http://www.dmtf.org/standards/published_documents/DSP1045_1.0.pdf

2706 DMTF DSP1047, *Storage Resource Virtualization Profile 1.0*
2707 http://www.dmtf.org/standards/published_documents/DSP1047_1.0.pdf

2708 DMTF DSP1022, *CPU Profile 1.0*,
2709 http://www.dmtf.org/standards/published_documents/DSP1022_1.0.pdf

2710 DMTF DSP1026, *System Memory Profile 1.0*,
2711 http://www.dmtf.org/standards/published_documents/DSP1026_1.0.pdf

2712 DMTF DSP1014, *Ethernet Port Profile 1.0*,
2713 http://www.dmtf.org/standards/published_documents/DSP1014_1.0.pdf

2714 DMTF DSP1050, *Ethernet Port Resource Virtualization Profile 1.1*
2715 http://www.dmtf.org/standards/published_documents/DSP1050_1.1.pdf

2716 DMTF DSP8049, *Network Port Profile XML Schema 1.0*
2717 http://schema.dmtf.org/ovf/networkportprofile/1/DSP8049_1.0.xsd

2718