

1 Devices Profile for Web Services

2 February 2006

3 Co-Developers

4 Shannon Chan, Microsoft

5 Dan Conti, Microsoft

6 Chris Kaler, Microsoft

7 Thomas Kuehnel, Microsoft

8 Alain Regnier, Ricoh

9 Bryan Roe, Intel

10 Dale Sather, Microsoft

11 Jeffrey Schlimmer, Microsoft (Editor)

12 Hitoshi Sekine, Ricoh

13 Jorgen Thelin, Microsoft (Editor)

14 Doug Walter, Microsoft

15 Jack Weast, Intel

16 Dave Whitehead, Lexmark

17 Don Wright, Lexmark

18 Yevgeniy Yarmosh, Intel

19 Copyright Notice

20 (c) 2004-2006 [Microsoft Corporation](#). All rights reserved.

21 Permission to copy and display the Devices Profile for Web Services (the "Profile"), in
22 any medium without fee or royalty is hereby granted, provided that you include the
23 following on ALL copies of the Profile, or portions thereof, that you make:

- 24 1. A link or URL to the Profile at this location.
- 25 2. The copyright notice as shown in the Profile.

26 THE PROFILE IS PROVIDED "AS IS," AND THE CO-DEVELOPERS MAKE NO
27 REPRESENTATIONS OR WARRANTIES, EXPRESS OR IMPLIED, INCLUDING, BUT NOT
28 LIMITED TO, WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
29 PURPOSE, NON-INFRINGEMENT, OR TITLE; THAT THE CONTENTS OF THE PROFILE
30 IS SUITABLE FOR ANY PURPOSE; NOR THAT THE IMPLEMENTATION OF SUCH
31 CONTENTS WILL NOT INFRINGE ANY THIRD PARTY PATENTS, COPYRIGHTS,
32 TRADEMARKS OR OTHER RIGHTS.

33 THE CO-DEVELOPERS WILL NOT BE LIABLE FOR ANY DIRECT, INDIRECT, SPECIAL,
34 INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF OR RELATING TO ANY
35 USE OR DISTRIBUTION OF THE PROFILE.

36 The name and trademarks of the Co-developers may NOT be used in any manner,
37 including advertising or publicity pertaining to the Profile or its contents without
38 specific, written prior permission. Title to copyright in the Profile will at all times
39 remain with Microsoft.

40 No other rights are granted by implication, estoppel or otherwise.

41	Abstract
42	This profile defines a minimal set of implementation constraints to enable secure
43	Web service messaging, discovery, description, and eventing on resource-
44	constrained endpoints.
45	Status
46	This is a public consultation draft release of this specification for community
47	evaluation and review. We welcome feedback on this specification through the WS-*
48	Workshop process.
49	Table of Contents
50	1. Introduction
51	1.1 Requirements
52	2. Terminology and Notation
53	2.1 Terminology
54	2.2 XML Namespaces
55	2.3 Notational Conventions
56	2.4 Compliance
57	3. Messaging
58	3.1 URI
59	3.2 UDP
60	3.3 HTTP
61	3.4 SOAP Envelope
62	3.5 WS-Addressing
63	3.6 Attachments
64	4. Discovery
65	5. Description
66	5.1 Characteristics
67	5.2 Hosting
68	5.3 WSDL
69	5.4 WS-Policy
70	6. Eventing
71	6.1 Subscription
72	6.2 Subscription Duration and Renewal
73	7. Security
74	7.1 Secure Communication
75	7.2 Security Policy Assertions

76 **8. Acknowledgements**
77 **9. References**
78 **10. Informative References**
79 **Appendix I – Constants**
80 **Appendix II – XML Schema**
81

82 **1. Introduction**

83 The Web services architecture includes a suite of specifications that define rich
84 functions and that may be composed to meet varied service requirements. To
85 promote both interoperability between resource-constrained Web service
86 implementations and interoperability with more flexible client implementations, this
87 profile identifies a core set of Web service specifications in the following areas:

- 88 • Sending secure messages to and from a Web service
- 89 • Dynamically discovering a Web service
- 90 • Describing a Web service
- 91 • Subscribing to, and receiving events from, a Web service

92 In each of these areas of scope, this profile defines minimal implementation
93 requirements for compliant Web service implementations.

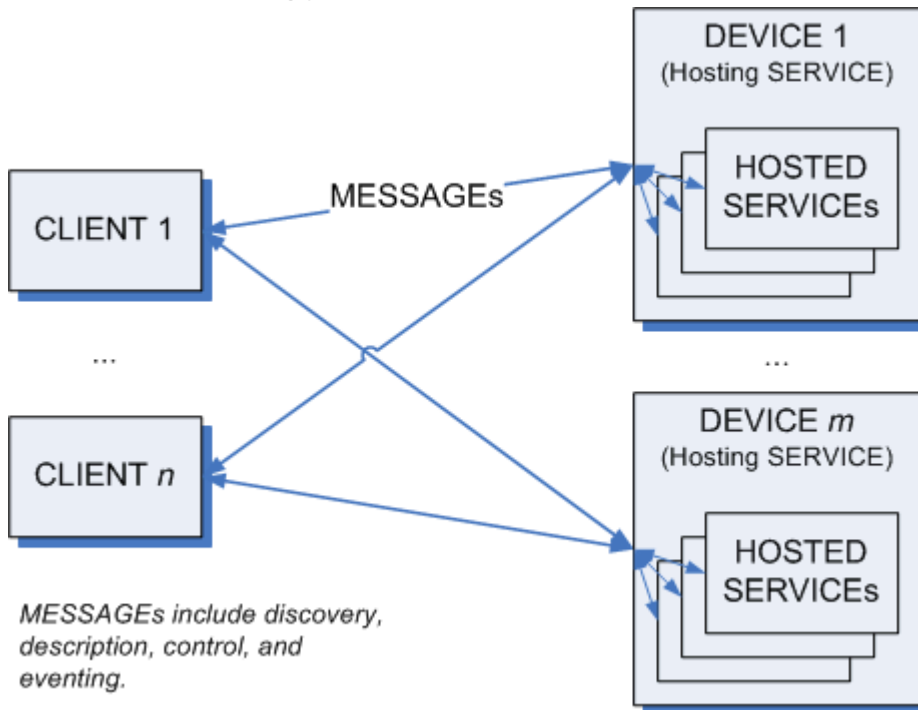
94 **1.1 Requirements**

95 This profile intends to meet the following requirements:

- 96 • Identify a minimal set of Web service specifications needed to enable secure
97 messaging, dynamic discovery, description, and eventing.
- 98 • Constrain Web services protocols and formats so Web services can be
99 implemented on peripheral-class and consumer electronics-class hardware.
- 100 • Define minimum requirements for compliance without constraining richer
101 implementations.

102 2. Terminology and Notation

103 2.1 Terminology



104

105 MESSAGE

106 Protocol elements that are exchanged, usually over a network, to affect a Web
107 service. Always includes a SOAP ENVELOPE. Typically also includes transport
108 framing information such as HTTP headers, TCP headers, and IP headers.

109 SOAP ENVELOPE

110 An XML Infoset that consists of a document information item [[XML Infoset](#)] with
111 exactly one member in its **[children]** property, which MUST be the SOAP
112 Envelope [[SOAP 1.2](#)] element information item.

113 MIME SOAP ENVELOPE

114 A SOAP ENVELOPE serialized using MIME Multipart Serialization [[MTOM](#)].

115 TEXT SOAP ENVELOPE

116 A SOAP ENVELOPE serialized as application/soap+xml.

117 CLIENT

118 A network endpoint that sends MESSAGES to and/or receives MESSAGES from a
119 SERVICE.

120 SERVICE

121 A network endpoint that receives and/or sends MESSAGES to provide a service.

122 DEVICE

123 A distinguished type of SERVICE that hosts other SERVICES and sends and/or
124 receives one or more specific types of MESSAGES.

125 HOSTED SERVICE

126 A distinguished type of SERVICE that is hosted by another SERVICE. The lifetime
127 of the HOSTED SERVICE is a subset of the lifetime of its host. The HOSTED

128 SERVICE is visible (not encapsulated) and is addressed separately from its host.
 129 Each HOSTED SERVICE has exactly one host. (The relationship is not transitive.)
 130 SENDER
 131 A CLIENT or SERVICE that sends a MESSAGE.
 132 RECEIVER
 133 A CLIENT or SERVICE that receives a MESSAGE.

134 2.2 XML Namespaces

135 The XML namespace URI that MUST be used by implementations of this specification
 136 is:

137 <http://schemas.xmlsoap.org/ws/2006/02/devprof>

138 Table 1 lists XML namespaces that are used in this specification. The choice of any
 139 namespace prefix is arbitrary and not semantically significant.

140 **Table 1: Prefixes and XML namespaces used in this specification.**

Prefix	XML Namespace	Specification(s)
soap	http://www.w3.org/2003/05/soap-envelope	[SOAP 1.2]
wsa	http://schemas.xmlsoap.org/ws/2004/08/addressing	[WS-Addressing]
wsd	http://schemas.xmlsoap.org/ws/2005/04/discovery	[WS-Discovery]
wsdp	http://schemas.xmlsoap.org/ws/2006/02/devprof	This profile
wsdl	http://schemas.xmlsoap.org/wsdl/	[WSDL 1.1]
wse	http://schemas.xmlsoap.org/ws/2004/08/eventing	[WS-Eventing]
wsoap	http://schemas.xmlsoap.org/wsdl/soap12/	[WSDL Binding for SOAP 1.2]
wsp	http://schemas.xmlsoap.org/ws/2004/09/policy	[WS-Policy, WS-PolicyAttachment]
wsu	http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd	[WS-Security 2004]
wsx	http://schemas.xmlsoap.org/ws/2004/09/mex	[WS-MetadataExchange]
xs	http://www.w3.org/2001/XMLSchema	[XML Schema Part 1, Part 2]

141 2.3 Notational Conventions

142 The keywords "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT",
 143 "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this
 144 document are to be interpreted as described in RFC 2119 [RFC 2119].

145 This specification uses the following syntax to define normative outlines for
 146 messages:

- 147 • The syntax appears as an XML instance, but values in italics indicate data types
 148 instead of literal values.
- 149 • Characters are appended to elements and attributes to indicate cardinality:

- 150 • "?" (0 or 1)
- 151 • "*" (0 or more)
- 152 • "+" (1 or more)
- 153 • The character "|" is used to indicate a choice between alternatives.
- 154 • The characters "(" and ")" are used to indicate that contained items are to be
155 treated as a group with respect to cardinality or choice.
- 156 • The characters "[" and "]" are used to call out references and property names.
- 157 • Ellipses (i.e., "...") indicate points of extensibility. Additional children and/or
158 attributes MAY be added at the indicated extension points but MUST NOT
159 contradict the semantics of the parent and/or owner, respectively. By default, if a
160 receiver does not recognize an extension, the receiver SHOULD ignore the
161 extension; exceptions to this processing rule, if any, are clearly indicated below.
- 162 • XML namespace prefixes (see Table 1) are used to indicate the namespace of the
163 element being defined.

164 This specification uses the **[action]** and Fault properties [[WS-Addressing](#)] to define
165 faults.

166 Normative statements in this profile are called out explicitly as follows:

167 *Rnnn: Normative statement text goes here.*

168 where "nnnn" is replaced by the statement number. Each statement contains exactly
169 one requirement level keyword (e.g., "MUST") and one conformance target keyword
170 (e.g., "MESSAGE").

171 2.4 Compliance

172 An endpoint MAY implement more than one of the roles defined herein. An endpoint
173 is not compliant with this specification if it fails to satisfy one or more of the MUST or
174 REQUIRED level requirements defined herein for the roles it implements.

175 Normative text within this specification takes precedence over normative outlines,
176 which in turn take precedence over the XML Schema [[XML Schema Part 1](#), [Part 2](#)]
177 descriptions, which in turn take precedence over examples.

178 3. Messaging

179 The scope of this section is the following set of Web services specifications. All of the
180 requirements in these specifications are included by reference except where
181 superseded by normative statements herein:

- 182 • [[SOAP 1.2, Part 1](#)]
- 183 • [[SOAP 1.2 Part 2, Section 7](#)]
- 184 • [[SOAP-over-UDP](#)]
- 185 • [[HTTP/1.1](#)]
- 186 • [[WS-Addressing](#)]
- 187 • [[RFC 4122](#)]
- 188 • [[MTOM](#)]

189 It is assumed that a DEVICE has obtained valid IPv4 and/or IPv6 addresses that do
190 not conflict with other addresses on the network. Mechanisms for obtaining IP

191 addresses are out of the scope of this profile. For more information, see [\[DHCP\]](#) and
192 [\[IPv6 Autoconfig\]](#).

193 3.1 URI

194 *R0025: A SERVICE MAY fail to process any URI with more than MAX_URI_SIZE*
195 *octets.*

196 *R0027: A SERVICE SHOULD NOT generate a URI with more than MAX_URI_SIZE*
197 *octets.*

198 The constant MAX_URI_SIZE is defined in Appendix I – Constants.

199 3.2 UDP

200 *R0029: A SERVICE SHOULD NOT send a SOAP ENVELOPE that has more octets than*
201 *the MTU over UDP.*

202 To improve reliability, a SERVICE should minimize the size of SOAP ENVELOPES sent
203 over UDP. However, some SOAP ENVELOPES may be larger than an MTU; for
204 example, a signed Hello SOAP ENVELOPE. If a SOAP ENVELOPE is larger than an
205 MTU, the underlying IP network stacks may fragment and reassemble the UDP
206 packet.

207 3.3 HTTP

208 *R0001: A SERVICE MUST support transfer-coding = "chunked".*

209 *R0012: A SERVICE MUST at least support the SOAP HTTP Binding.*

210 *R0013: A SERVICE MUST at least implement the Responding SOAP Node of the SOAP*
211 *Request-Response Message Exchange Pattern*
212 *(<http://www.w3.org/2003/05/soap/mep/request-response/>).*

213 *R0014: A SERVICE MAY choose not to implement the Responding SOAP Node of the*
214 *SOAP Response Message Exchange Pattern*
215 *(<http://www.w3.org/2003/05/soap/mep/soap-response/>).*

216 *R0015: A SERVICE MAY choose not to support the SOAP Web Method Feature.*

217 R0014 and R0015 relax requirements in [\[SOAP 1.2, Part 2, Section 7\]](#).

218 *R0030: A SERVICE MUST at least implement the Responding SOAP Node of an HTTP*
219 *one-way Message Exchange Pattern where the SOAP ENVELOPE is carried in*
220 *the HTTP Request and the HTTP Response has a Status Code of 202 Accepted*
221 *and an empty Entity Body (no SOAP ENVELOPE).*

222 *R0017: A SERVICE MUST at least support Request Message SOAP ENVELOPES and*
223 *one-way SOAP ENVELOPES that are delivered using HTTP POST.*

224 3.4 SOAP Envelope

225 *R0034: A SERVICE MUST at least receive and send SOAP 1.2 [\[SOAP 1.2\]](#) SOAP*
226 *ENVELOPES.*

227 *R0003: A SERVICE MAY reject a TEXT SOAP ENVELOPE with more than*
228 *MAX_ENVELOPE_SIZE octets.*

229 *R0026: A SERVICE SHOULD NOT send a TEXT SOAP ENVELOPE with more than*
230 *MAX_ENVELOPE_SIZE octets.*

231 Large SOAP ENVELOPES are expected to be serialized using attachments.

232 3.5 WS-Addressing

233 *R0004: A DEVICE SHOULD use a urn:uuid scheme URI as the **[address]** property of*
234 *its Endpoint Reference.*

235 *R0005: A DEVICE MUST use a stable, globally unique identifier that is constant*
236 *across network interfaces and IPv4/v6 addresses as the **[address]** property*
237 *of its Endpoint Reference.*

238 *R0006: A DEVICE MUST persist the **[address]** property of its Endpoint Reference*
239 *across re-initialization and changes in the metadata of the DEVICE and any*
240 *SERVICES it hosts.*

241 Because the **[address]** property of an Endpoint Reference [\[WS-Addressing\]](#) is a
242 SOAP-layer address, there is no requirement to use anything other than a UUID for
243 the **[address]** property.

244 *R0007: A DEVICE SHOULD NOT include any **[reference property]** properties in its*
245 *Endpoint Reference.*

246 The combination of the **[address]** and **[reference property]** properties defines the
247 identity of an Endpoint Reference. If the **[address]** property provides sufficient
248 identity information, there is no requirement to use **[reference property]**
249 properties to provide additional identity.

250 *R0042: A HOSTED SERVICE SHOULD use an HTTP transport address as the*
251 ***[address]** property of its Endpoint Reference.*

252 Use of other possible values of **[address]** by a HOSTED SERVICE is out of scope of
253 this profile.

254 *R0031: A SERVICE MUST generate a wsa:InvalidMessageInformationHeader SOAP*
255 *Fault if the **[address]** of the **[reply endpoint]** of an HTTP Request Message*
256 *SOAP ENVELOPE is not*
257 *"http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous".*

258 *R0041: If an HTTP Request Message SOAP ENVELOPE generates a SOAP Fault, a*
259 *SERVICE MAY discard the SOAP Fault if the **[address]** of the **[fault***
260 ***endpoint]** of the HTTP Request Message is not*
261 *"http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous".*

262 The SOAP HTTP Binding requires the Response Message SOAP ENVELOPE to be
263 transmitted as the HTTP Response of the corresponding Request Message SOAP
264 ENVELOPE.

265 *R0019: A SERVICE MUST include a Message Information Header representing a*
266 ***[relationship]** property of type wsa:Reply in each Response Message SOAP*
267 *ENVELOPE the service generates.*

268 Per WS-Addressing [\[WS-Addressing\]](#), a response SOAP ENVELOPE must include a
269 wsa:RelatesTo SOAP ENVELOPE header block. Since wsa:Reply is the default value
270 for the **[relationship]** property, the RelationshipType attribute should be omitted
271 from the wsa:RelatesTo SOAP ENVELOPE header block.

272 *R0040: A SERVICE MUST include a Message Information Header representing a*
273 ***[relationship]** property of type wsa:Reply in each SOAP Fault SOAP*
274 *ENVELOPE the service generates.*

275 **3.6 Attachments**

276 *R0022: If a SERVICE supports attachments, the SERVICE MUST support the HTTP*
277 *Transmission Optimization Feature.*

278 The HTTP Transmission Optimization Feature implies support for the Optimized MIME
279 Multipart Serialization and Abstract Transmission Optimization features.

280 *R0036: A SERVICE MAY reject a MIME SOAP ENVELOPE if the Content-Transfer-*
281 *Encoding header field mechanism of any MIME part is not "binary".*

282 *R0037: A SERVICE MUST NOT send a MIME SOAP ENVELOPE unless the Content-*
283 *Transfer-Encoding header field mechanism of every MIME part is "binary".*

284 Even for the SOAP Envelope, the "binary" Content-Transfer-Encoding mechanism is
285 more appropriate than the "8bit" mechanism which is suitable only for data that may
286 be represented as relatively short lines of at most 998 octets [MIME].

287 *R0038: A SERVICE MAY reject a MIME SOAP ENVELOPE if the root part is not the first*
288 *body part in the Multipart/Related entity.*

289 *R0039: A SERVICE MUST NOT send a MIME SOAP ENVELOPE unless root part is the*
290 *first body part in the Multipart/Related entity.*

291 Per MTOM, the root part of the MIME SOAP ENVELOPE contains an XML
292 representation of the modified SOAP Envelope, with additional parts that contain
293 binary representations of each attachment. This root part must be the first part so a
294 RECEIVER does not have to buffer attachments.

295 **4. Discovery**

296 The scope of this section is the following set of Web services specifications. All of the
297 requirements in these specifications are included by reference except where
298 superseded by normative statements herein:

- 299 • [WS-Discovery]

300 If a CLIENT and a SERVICE are not on the same subnet, the CLIENT may not be able
301 to discover the SERVICE. However, if a CLIENT has an Endpoint Reference and
302 transport address for a SERVICE through some other means, the CLIENT and
303 SERVICE should be able to communicate within the scope of this profile.

304 *R1013: A DEVICE MUST be a compliant Target Service.*

305 *R1001: A HOSTED SERVICE SHOULD NOT be a Target Service.*

306 If each SERVICE were to participate in WS-Discovery, the network traffic generated
307 by a relatively small number of DEVICES hosting a relatively small number of
308 HOSTED SERVICES could overwhelm a bandwidth-limited network. Therefore, only
309 DEVICES act as Target Services.

310 *R1019: A DEVICE MUST at least support the*
311 *"http://schemas.xmlsoap.org/ws/2005/04/discovery/rfc2396" and*
312 *"http://schemas.xmlsoap.org/ws/2005/04/discovery/strcmp0" Scope*
313 *matching rules.*

314 *R1020: If a DEVICE includes Types in a Hello, Probe Match, or Resolve Match SOAP*
315 *ENVELOPE, it MUST include the wsdp:Device Type.*

316 Including the wsdp:Device Type indicates a DEVICE supports the Devices Profile,
317 including allowing the retrieving metadata about the DEVICE and any HOSTED
318 SERVICES using Get [[WS-Transfer](#)].

319 *R1009: A DEVICE MUST at least support receiving Probe and Resolve SOAP*
320 *ENVELOPEs and sending Hello and Bye SOAP ENVELOPEs over multicast UDP.*

321 *R1016: A DEVICE MUST at least support sending Probe Match and Resolve Match*
322 *SOAP ENVELOPEs over unicast UDP.*

323 *R1018: A DEVICE MAY ignore a multicast UDP Probe or Resolve SOAP ENVELOPE if*
324 *the **[address]** of the **[reply endpoint]** is not*
325 *"http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous".*

326 WS-Discovery acknowledges that a CLIENT may include reply information in UDP
327 Probe and Resolve SOAP ENVELOPEs to specify a transport other than SOAP over
328 UDP. However, to establish a baseline for interoperability, DEVICES are required only
329 to support UDP responses.

330 *R1015: A DEVICE MUST support receiving a Probe SOAP ENVELOPE as an HTTP*
331 *Request.*

332 *R1021: If a DEVICE matches a Probe SOAP ENVELOPE received as an HTTP Request,*
333 *it MUST send a Probe Match SOAP ENVELOPE in the HTTP Response.*

334 *R1022: If a DEVICE does not match a Probe SOAP ENVELOPE received as an HTTP*
335 *Request, it MUST send an HTTP Response with a Status Code of 202 Accepted*
336 *and an empty Entity Body (no SOAP ENVELOPE).*

337 To support the scenario where a DEVICE has a known HTTP address, a CLIENT may
338 send a Probe over HTTP to that address and expect to receive either a Probe Match
339 (if the Probe matches the DEVICE listening on that address) or an empty HTTP
340 Response (otherwise).

341 5. Description

342 The scope of this section is the following set of Web services specifications. All of the
343 requirements in these specifications are included by reference except where
344 superseded by normative statements herein:

- 345 • [[XML Schema Part 1, Part 2](#)]
- 346 • [[WSDL 1.1](#)]
- 347 • [[BP 1.1, Section 4](#)]
- 348 • [[WSDL Binding for SOAP 1.2](#)]
- 349 • [[WS-MetadataExchange](#)]
- 350 • [[WS-Policy](#)]
- 351 • [[WS-PolicyAttachment](#)]
- 352 • [[WS-Transfer](#)]

353 In highly-constrained circumstances, a CLIENT will know all it needs to know about a
354 DEVICE and its HOSTED SERVICES to correctly send and receive application-specific
355 MESSAGES. However, in development scenarios, or when a CLIENT wishes to inspect
356 a DEVICE and take advantage of extended or nonstandard capabilities, a CLIENT will

357 need to retrieve the description (a.k.a. metadata) for a DEVICE and/or its HOSTED
358 SERVICES.

359 The description for a DEVICE is retrieved by sending a WS-Transfer Get SOAP
360 ENVELOPE to the DEVICE. The description conveys generic DEVICE characteristics
361 and may be extended to convey domain-specific SERVICE characteristics. Description
362 also indicates which HOSTED SERVICES are hosted by a DEVICE; in many
363 circumstances, a CLIENT will need to retrieve the description for one or more
364 HOSTED SERVICES as well as for the DEVICE.

365 Through WSDL, description also conveys the MESSAGES a HOSTED SERVICE is
366 capable of receiving and sending. Through WS-Policy, description conveys the
367 capabilities and requirements of a HOSTED SERVICE, particularly the transports over
368 which it may be reached and its security capabilities.

369 *R2044: In a Get Response SOAP ENVELOPE, A DEVICE MUST include only a*
370 *wsx:Metadata element in the SOAP ENVELOPE Body.*

371 All metadata from the device should be contained in the wsx:Metadata element in
372 the Get Response.

373 *R2045: A DEVICE MAY generate a wsa:ActionNotSupported SOAP Fault in response*
374 *to a Put, Delete, or Create SOAP ENVELOPE.*

375 A DEVICE is not required to support all of the operations defined in [\[WS-Transfer\]](#).

376 5.1 Characteristics

377 To express DEVICE characteristics that are typically fixed across all DEVICES of the
378 same model by their manufacturer, this profile defines extensible ThisModel
379 metadata as follows:

```
380 <wsdp:ThisModel ...>  
381   <wsdp:Manufacturer xml:lang="..."? >xs:string</wsdp:Manufacturer>+  
382   <wsdp:ManufacturerUrl>xs:anyURI</wsdp:ManufacturerUrl?>  
383   <wsdp:ModelName xml:lang="..."? >xs:string</wsdp:ModelName>+  
384   <wsdp:ModelNumber>xs:string</wsdp:ModelNumber?>  
385   <wsdp:ModelUrl>xs:anyURI</wsdp:ModelUrl?>  
386   <wsdp:PresentationUrl>xs:anyURI</wsdp:PresentationUrl?>  
387   ...  
388 </wsdp:ThisModel>
```

389 The following describes additional, normative constraints on the outline above:

390 wsdp:ThisModel/ wsdp:Manufacturer

391 Name of the manufacturer of the DEVICE. It MUST have fewer than
392 MAX_FIELD_SIZE Unicode characters, SHOULD be localized, and SHOULD be
393 repeated for each supported locale.

394 wsdp:ThisModel/ wsdp:ManufacturerUrl

395 URL to a Web site for the manufacturer of the DEVICE. It MUST have fewer than
396 MAX_URI_SIZE octets.

397 wsdp:ThisModel/ wsdp:ModelName

398 User-friendly name for this model of device chosen by the manufacturer. It MUST
399 have fewer than MAX_FIELD_SIZE Unicode characters, SHOULD be localized, and
400 SHOULD be repeated for each supported locale.

401 wsdp:ThisModel/ wsdp:ModelNumber

402 Model number for this model of DEVICE. It MUST have fewer than
403 MAX_FIELD_SIZE Unicode characters.
404 wsdp:ThisModel/ wsdp:ModelUrl
405 URL to a Web site for this model of DEVICE. It MUST have fewer than
406 MAX_URI_SIZE octets.
407 wsdp:ThisModel/ wsdp:PresentationUrl
408 URL to an HTML page for this DEVICE. It MAY be relative to a base URL and MUST
409 have fewer than MAX_URI_SIZE octets.

410 CORRECT:

```
411 <wsdp:ThisModel  
412   xmlns:wsdp="http://schemas.xmlsoap.org/ws/2006/02/devprof" >  
413   <wsdp:Manufacturer>ACME Manufacturing</wsdp:Manufacturer>  
414   <wsdp:ModelName xml:lang="en-GB" >ColourBeam 9</wsdp:ModelName>  
415   <wsdp:ModelName xml:lang="en-US" >ColorBeam 9</wsdp:ModelName>  
416 </wsdp:ThisModel>  
417
```

418 A Dialect [[WS-MetadataExchange](#)] equal to
419 "http://schemas.xmlsoap.org/ws/2006/02/devprof/ThisModel" indicates an instance
420 of the ThisModel metadata format.

421 No Identifier [[WS-MetadataExchange](#)] is defined for instances of the ThisModel
422 metadata format.

423 *R2038: A DEVICE MUST have one Metadata Section with Dialect equal to*
424 *"http://schemas.xmlsoap.org/ws/2006/02/devprof/ThisModel" for its*
425 *ThisModel metadata.*

426 *R2012: In any Get Response SOAP ENVELOPE, a DEVICE MUST include the Metadata*
427 *Section with Dialect equal to*
428 *"http://schemas.xmlsoap.org/ws/2006/02/devprof/ThisModel".*

429 Get [[WS-Transfer](#)] is the interoperable means for a CLIENT to retrieve the resource
430 representation data for a DEVICE – which includes the ThisModel metadata for a
431 DEVICE. A DEVICE may also provide other means for a CLIENT to retrieve its
432 ThisModel metadata.

433 *R2001: If a DEVICE changes any of its ThisModel metadata, it MUST increment the*
434 *Metadata Version exposed in Hello, Probe Match, and Resolve Match SOAP*
435 *ENVELOPEs as wsdl:MetadataVersion.*

436 Caching for the ThisModel metadata is controlled by the wsdl:MetadataVersion
437 construct [[WS-Discovery](#)].

438 To express DEVICE characteristics that typically vary from one DEVICE to another of
439 the same kind, this profile defines extensible ThisDevice metadata as follows:

```
440 <wsdp:ThisDevice ...>  
441   <wsdp:FriendlyName xml:lang="..."? >xs:string</wsdp:FriendlyName>+  
442   <wsdp:FirmwareVersion>xs:string</wsdp:FirmwareVersion?>  
443   <wsdp:SerialNumber>xs:string</wsdp:SerialNumber?>  
444   ...  
445 </wsdp:ThisDevice>
```

446 The following describes additional, normative constraints on the outline above:

447 wsdp:ThisDevice/ wsdp:FriendlyName

448 User-friendly name for this DEVICE. It MUST have fewer than MAX_FIELD_SIZE
449 Unicode characters, SHOULD be localized, and SHOULD be repeated for each
450 supported locale.

451 wsdp:ThisDevice/ wsdp:FirmwareVersion
452 Firmware version for this DEVICE. It MUST have fewer than MAX_FIELD_SIZE
453 Unicode characters.

454 wsdp:ThisDevice/ wsdp:SerialNumber
455 Manufacturer-assigned serial number for this DEVICE. It MUST have fewer than
456 MAX_FIELD_SIZE Unicode characters.

457 CORRECT:

```
458 <wsdp:ThisDevice  
459     xmlns:wsdp="http://schemas.xmlsoap.org/ws/2006/02/devprof" >  
460   <wsdp:FriendlyName xml:lang="en-GB" >  
461     ACME ColourBeam Printer  
462   </wsdp:FriendlyName>  
463   <wsdp:FriendlyName xml:lang="en-US" >  
464     ACME ColorBeam Printer  
465   </wsdp:FriendlyName>  
466 </wsdp:ThisDevice>  
467
```

468 A Dialect [[WS-MetadataExchange](#)] equal to
469 "http://schemas.xmlsoap.org/ws/2006/02/devprof/ThisDevice" indicates an instance
470 of the ThisDevice metadata format.

471 No Identifier [[WS-MetadataExchange](#)] is defined for instances of the ThisDevice
472 metadata format.

473 *R2039: A DEVICE MUST have a Metadata Section with Dialect equal to
474 "http://schemas.xmlsoap.org/ws/2006/02/devprof/ThisDevice" for its
475 ThisDevice metadata.*

476 *R2014: In any Get Response SOAP ENVELOPE, a DEVICE MUST include the Metadata
477 Section with Dialect equal to
478 "http://schemas.xmlsoap.org/ws/2006/02/devprof/ThisDevice".*

479 CORRECT:

```
480 <soap:Envelope  
481     xmlns:soap="http://www.w3.org/2003/05/soap-envelope"  
482     xmlns:wsdp="http://schemas.xmlsoap.org/ws/2006/02/devprof"  
483     xmlns:wsx="http://schemas.xmlsoap.org/ws/2004/09/mex"  
484     xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing" >  
485   <soap:Header>  
486     <wsa:Action>  
487       http://schemas.xmlsoap.org/ws/2004/09/transfer/GetResponse  
488     </wsa:Action>  
489     <wsa:RelatesTo>  
490       urn:uuid:82204a83-52f6-475c-9708-174fa27659ec  
491     </wsa:RelatesTo>  
492     <wsa:To>  
493       http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous  
494     </wsa:To>  
495   </soap:Header>  
496   <soap:Body>  
497     <wsx:Metadata>
```

```

498
499     <wsx:MetadataSection
500 Dialect="http://schemas.xmlsoap.org/ws/2006/02/devprof/ThisModel "
501 >
502     <wsdp:ThisModel>
503         <wsdp:Manufacturer>ACME Manufacturing</wsdp:Manufacturer>
504         <wsdp:ModelName xml:lang="en-GB" >
505             ColourBeam 9
506         </wsdp:ModelName>
507         <wsdp:ModelName xml:lang="en-US" >
508             ColorBeam 9
509         </wsdp:ModelName>
510     </wsdp:ThisModel>
511 </wsx:MetadataSection>
512
513     <wsx:MetadataSection
514 Dialect="http://schemas.xmlsoap.org/ws/2006/02/devprof/ThisDevice "
515 >
516     <wsdp:ThisDevice>
517         <wsdp:FriendlyName xml:lang="en-GB" >
518             ACME ColourBeam Printer
519         </wsdp:FriendlyName>
520         <wsdp:FriendlyName xml:lang="en-US" >
521             ACME ColorBeam Printer
522         </wsdp:FriendlyName>
523     </wsdp:ThisDevice>
524 </wsx:MetadataSection>
525
526     <!-- Other Metadata Sections omitted for brevity. -->
527
528 </wsx:Metadata>
529 </soap:Body>
530 </soap:Envelope>
531

```

532 Get [[WS-Transfer](#)] is the interoperable means for a CLIENT to retrieve the resource
533 representation data for a DEVICE – which includes the ThisDevice metadata for a
534 DEVICE. A DEVICE may also provide other means for a CLIENT to retrieve its
535 ThisDevice metadata.

536 *R2002: If a DEVICE changes any of its ThisDevice metadata, it MUST increment the*
537 *Metadata Version exposed in Hello, Probe Match, and Resolve Match SOAP*
538 *ENVELOPES as wsdl:MetadataVersion.*

539 Caching for the ThisDevice metadata is controlled by the wsdl:MetadataVersion
540 construct [[WS-Discovery](#)].

541 5.2 Hosting

542 To express the relationship between a HOSTED SERVICE and its host, this profile
543 defines relationship metadata as follows:

```

544 <wsdp:Relationship Type="xs:anyURI" ... >
545     (<wsdp:Host>
546         <wsa:EndpointReference>endpoint-reference</wsa:EndpointReference>+
547         <wsdp:Types>list of xs:QName</wsdp:Types>?
548         <wsdp:ServiceId>xs:anyURI</wsdp:ServiceId>
549         ...

```



```

550 </wsdp:Host>)?
551 (<wsdp:Hosted>
552   <wsa:EndpointReference>endpoint-reference</wsa:EndpointReference>+
553   <wsdp:Types>list of xs:QName</wsdp:Types>?
554   <wsdp:ServiceId>xs:anyURI</wsdp:ServiceId>
555   ...
556 </wsdp:Hosted>)*
557 ...
558 </wsdp:Relationship>

```

559 The following describes additional, normative constraints on the outline above:

560 wsdp:Relationship

561 This is a general mechanism for defining a relationship between two or more
562 SERVICES.

563 wsdp:Relationship/@Type

564 The type of the relationship. The nature of the relationship and the content of the
565 wsdp:Relationship element are determined by this value. This value should be
566 compared directly, as a case-sensitive string, with no attempt to make a relative
567 URI into an absolute URI, to unescape, or to otherwise canonicalize it.

568 wsdp:Relationship/@Type =

569 "http://schemas.xmlsoap.org/ws/2006/02/devprof/host"

570 This is a specific, hosting relationship type to indicate the relationship between a
571 HOSTED SERVICE and its host. This relationship type defines the following
572 additional content:

573 wsdp:Relationship/wsdp:Host

574 Endpoint References for the host. If omitted, implied value is the Endpoint
575 Reference of the SERVICE that returned this metadata in a Get Response SOAP
576 ENVELOPE. At least one of ./wsdp:Host or ./wsdp:Hosted MUST be included.

577 wsdp:Relationship/wsdp:Host/wsdp:Types

578 Unordered set of Types implemented by the host. (See [\[WS-Discovery\]](#).) If
579 omitted, no implied value.

580 The Types element is explicitly copied from the WS-Discovery XML namespace
581 into this one to make the XML Schema deterministic. Reusing the wsdl:Types
582 element from WS-Discovery would introduce non-determinism because there
583 would be an optional element from another XML namespace (wsdl:Types),
584 followed by an optional element (wsdp:ServiceId) and an optional wildcard for
585 elements from other XML namespaces.

586 wsdp:Relationship/wsdp:Host/wsdp:ServiceId

587 Identifier for the host which MUST be persisted across re-initialization (see also
588 [R0005](#) and [R0006](#)) and MUST NOT be shared across multiple Host elements. This
589 value should be compared directly, as a case-sensitive string, with no attempt to
590 make a relative URI into an absolute URI, to unescape, or to otherwise
591 canonicalize it.

592 wsdp:Relationship/wsdp:Hosted

593 Endpoint References for a HOSTED SERVICE. If omitted, implied value is the
594 Endpoint Reference of the SERVICE that returned this metadata in a Get
595 Response SOAP ENVELOPE. At least one of ./wsdp:Host or ./wsdp:Hosted MUST
596 be included.

597 For the hosting relationship type, if a host has more than one HOSTED SERVICE,
598 including one relationship that lists all HOSTED SERVICES is equivalent to
599 including multiple relationships that each list some subset of the HOSTED
600 SERVICES.

601 wsdp:Relationship/wsdp:Hosted/wsdp:Types

602 Unordered set of Types implemented by a HOSTED SERVICE. (See [\[WS-
603 Discovery\]](#).) If omitted, no implied value.

604 wsdp:Relationship/wsdp:Hosted/wsdp:ServiceId

605 Identifier for a HOSTED SERVICE which MUST be persisted across re-initialization
606 and MUST NOT be shared across multiple Hosted elements. ServiceId MUST be
607 unique within a DEVICE. This value should be compared directly, as a case-
608 sensitive string, with no attempt to make a relative URI into an absolute URI, to
609 unescape, or to otherwise canonicalize it.

610 CORRECT:

```
611 <wsdp:Relationship  
612   Type="http://schemas.xmlsoap.org/ws/2006/02/devprof/host "  
613   xmlns:img="http://printer.example.org/imaging"  
614   xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing"  
615   xmlns:wsdp="http://schemas.xmlsoap.org/ws/2006/02/devprof" >  
616   <wsdp:Hosted>  
617     <wsa:EndpointReference>  
618       <wsa:Address>http://172.30.184.244/print</wsa:Address>  
619     </wsa:EndpointReference>  
620     <wsdp:Types>  
621       img:PrintBasicPortType img:PrintAdvancedPortType  
622     </wsdp:Types>  
623     <wsdp:ServiceId>  
624       http://printer.example.org/imaging/PrintService  
625     </wsdp:ServiceId>  
626   </wsdp:Hosted>  
627 </wsdp:Relationship>  
628
```

629 A Dialect [\[WS-MetadataExchange\]](#) equal to

630 "http://schemas.xmlsoap.org/ws/2006/02/devprof/Relationship" indicates an
631 instance of the Relationship metadata format.

632 No Identifier [\[WS-MetadataExchange\]](#) is defined for instances of the Relationship
633 metadata format.

634 *R2040: If a SERVICE has any HOSTED SERVICES, it MUST have at least one
635 Metadata Section with Dialect equal to
636 "http://schemas.xmlsoap.org/ws/2006/02/devprof/Relationship" for its
637 Relationship metadata.*

638 *R2029: In any Get Response SOAP ENVELOPE, a SERVICE MUST include any
639 Metadata Section(s) with Dialect equal to
640 "http://schemas.xmlsoap.org/ws/2006/02/devprof/Relationship".*

641 Get [\[WS-Transfer\]](#) is the interoperable means for a CLIENT to retrieve the resource
642 representation data for a SERVICE – which includes the relationship metadata for a
643 SERVICE. A SERVICE may provide other means for a CLIENT to retrieve its
644 relationship metadata.

645 CORRECT:


```

646 <soap:Envelope
647   xmlns:gen="http://example.org/general"
648   xmlns:img="http://printer.example.org/imaging"
649   xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
650   xmlns:wsp="http://schemas.xmlsoap.org/ws/2006/02/devprof"
651   xmlns:wsx="http://schemas.xmlsoap.org/ws/2004/09/mex"
652   xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing" >
653 <soap:Header>
654   <wsa:Action>
655     http://schemas.xmlsoap.org/ws/2004/09/transfer/GetResponse
656   </wsa:Action>
657   <wsa:RelatesTo>
658     urn:uuid:82204a83-52f6-475c-9708-174fa27659ec
659   </wsa:RelatesTo>
660   <wsa:To>
661     http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous
662   </wsa:To>
663 </soap:Header>
664 <soap:Body>
665   <wsx:Metadata>
666     <wsx:MetadataSection
667       Dialect
668       ="http://schemas.xmlsoap.org/ws/2006/02/devprof/Relationship"
669     >
670       <wsp:Relationship
671         Type="http://schemas.xmlsoap.org/ws/2006/02/devprof/host" >
672         <wsp:Hosted>
673           <wsa:EndpointReference>
674             <wsa:Address>http://172.30.184.244/print</wsa:Address>
675           </wsa:EndpointReference>
676           <wsa:EndpointReference>
677             <wsa:Address>http://[fdaa:23]/print1</wsa:Address>
678           </wsa:EndpointReference>
679           <wsp:Types>
680             img:PrintBasicPortType img:PrintAdvancedPortType
681           </wsp:Types>
682           <wsp:ServiceId>
683             http://printer.example.org/imaging/PrintService
684           </wsp:ServiceId>
685         </wsp:Hosted>
686
687         <wsp:Hosted>
688           <wsa:EndpointReference>
689             <wsa:Address>http://172.30.184.244/scan</wsa:Address>
690           </wsa:EndpointReference>
691           <wsa:EndpointReference>
692             <wsa:Address>http://[fdaa:24]/scan</wsa:Address>
693           </wsa:EndpointReference>
694           <wsp:Types>img:ScanBasicPortType</wsp:Types>
695           <wsp:ServiceId>
696             http://printer.example.org/imaging/ScanService
697           </wsp:ServiceId>
698         </wsp:Hosted>
699       </wsp:Relationship>
700     </wsx:MetadataSection>
701
702     <!-- Other Metadata Sections omitted for brevity. -->

```

```
703 </wsx:Metadata>
704 </soap:Body>
705 </soap:Envelope>
706
707
```

708 *R2030: If a DEVICE changes any of its relationship metadata, it MUST increment the*
709 *Metadata Version exposed in Hello, Probe Match, and Resolve Match SOAP*
710 *ENVELOPES as wsdl:MetadataVersion.*

711 Caching for relationship metadata is controlled by the wsdl:MetadataVersion
712 construct [[WS-Discovery](#)].

713 *R2042: A DEVICE MUST NOT change its relationship metadata based on temporary*
714 *changes in the network availability of the SERVICES described by the*
715 *metadata.*

716 Relationship metadata is intended to model fairly static relationships and should not
717 change if a SERVICE becomes temporarily unavailable. As in the general case, any
718 CLIENT attempting to contact such a SERVICE will need to deal with an Endpoint
719 Unavailable Fault [[WS-Addressing](#)], connection refusal, or other network indication
720 that the SERVICE is unavailable.

721 5.3 WSDL

722 *R2004: If a HOSTED SERVICE exposes Notifications, its portType MUST include*
723 *Notification and/or Solicit-Response Operations describing those Notifications.*

724 R2004 relaxes R2303 in [[BP 1.1, Section 4](#)].

725 *R2019: A HOSTED SERVICE MUST at least include a document-literal Binding for*
726 *each portType in its WSDL.*

727 Because the document-literal SOAP Binding is more general than an rpc-literal SOAP
728 Binding, there is no requirement to use anything other than the document-literal
729 Binding.

730 *R2020: A HOSTED SERVICE MUST at least include a WSDL Binding for SOAP 1.2 for*
731 *each portType in its WSDL.*

732 *R2028: A HOSTED SERVICE is not required to include any WSDL bindings for SOAP*
733 *1.1 in its WSDL.*

734 Since this profile brings SOAP 1.2 into scope, it is sufficient to bind to that version of
735 SOAP. There is no requirement to bind to other SOAP versions and thus R2028
736 updates R2401 in [[BP 1.1, Section 4](#)] to SOAP 1.2.

737 *R2043: A HOSTED SERVICE is not required to include any WSDL Services in its*
738 *WSDL.*

739 Since addressing information for a HOSTED SERVICE is included in relationship
740 metadata, there is no requirement to re-express this information in WSDL Service(s)
741 or Port(s).

742 *R2023: If a HOSTED SERVICE receives a MESSAGE that is inconsistent with its WSDL*
743 *description, the HOSTED SERVICE SHOULD generate a SOAP Fault with a*
744 *Code Value of "Sender", unless a "MustUnderstand" or "VersionMismatch"*
745 *Fault is generated.*

746 *R2024: If a HOSTED SERVICE receives a MESSAGE that is inconsistent with its WSDL*
747 *description, the HOSTED SERVICE MUST check for "VersionMismatch",*
748 *"MustUnderstand", and "Sender" fault conditions in that order.*

749 Statements R2023 and R2024 update R2724 and R2725 [BP 1.1, Section 4] to SOAP
750 1.2.

751 *R2031: A HOSTED SERVICE MUST have at least one Metadata Section with*
752 *Dialect="http://schemas.xmlsoap.org/wsdl/".*

753 For clarity, separation of levels of abstraction, and/or reuse of standardized
754 components, WSDL may be authored in a style that separates different elements of a
755 Service Definition into separate documents which may be imported or included as
756 needed. Each separate document may be available at the URL in the
757 xs:include/@schemaLocation, xs:import/@schemaLocation, or wsdl:import/@location
758 or may be included in a separate XML Schema or WSDL Metadata Section.

759 *R2016: In any Get Response SOAP ENVELOPE, a HOSTED SERVICE MUST include the*
760 *Metadata Section(s) with Dialect equal to*
761 *"http://schemas.xmlsoap.org/wsdl/".*

762 Get [WS-Transfer] is the interoperable means for a CLIENT to retrieve resource
763 representation data for a HOSTED SERVICE – which includes the WSDL for a
764 HOSTED SERVICE. A HOSTED SERVICE may provide other means for a CLIENT to
765 retrieve its WSDL.

766 There is no requirement for a HOSTED SERVICE to store its WSDL and include it in-
767 line in a Get Response SOAP ENVELOPE. The WSDL may be stored at a different
768 location, and the HOSTED SERVICE may include a reference to it in a Get Response
769 SOAP ENVELOPE.

770 CORRECT:

```
771 <soap:Envelope  
772     xmlns:soap="http://www.w3.org/2003/05/soap-envelope"  
773     xmlns:wsx="http://schemas.xmlsoap.org/ws/2004/09/mex"  
774     xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing" >  
775   <soap:Header>  
776     <wsa:Action>  
777       http://schemas.xmlsoap.org/ws/2004/09/transfer/GetResponse  
778     </wsa:Action>  
779     <wsa:RelatesTo>  
780       urn:uuid:82204a83-52f6-475c-9708-174fa27659ec  
781     </wsa:RelatesTo>  
782     <wsa:To>  
783       http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous  
784     </wsa:To>  
785   </soap:Header>  
786   <soap:Body>  
787     <wsx:Metadata>  
788  
789     <wsx:MetadataSection  
790       Dialect="http://schemas.xmlsoap.org/wsdl" >  
791     <wsx:MetadataReference>  
792       <wsa:Address>http://172.30.184.244/print</wsa:Address>  
793       <wsa:ReferenceParameters>  
794         <x:Acme xmlns:x="urn:acme.com:webservices">  
795           WSDL
```

```

796         </x:Acme>
797         </wsa:ReferenceParameters>
798         </wsx:MetadataReference>
799         </wsx:MetadataSection>
800
801         <!-- Other Metadata Sections omitted for brevity. -->
802
803     </wsx:Metadata>
804 </soap:Body>
805 </soap:Envelope>
806

```

807 5.4 WS-Policy

808 To indicate that a DEVICE is compliant with this profile, this profile defines the
809 following WS-Policy [[WS-Policy](#)] assertion:

```
810 <wsdp:Profile wsp:Optional="true"? ... />
```

811 The following describes additional, normative constraints on the outline above:

812 wsdp:Profile

813 Assertion indicating compliance with this profile is required. This assertion has
814 Endpoint Policy Subject [[WS-PolicyAttachment](#)]: a policy expression containing
815 this assertion MAY be attached to a wsdl:port, SHOULD be attached to a
816 wsdl:binding, but MUST NOT be attached to a wsdl:portType; the latter is
817 prohibited because the assertion specifies a concrete behavior whereas the
818 wsdl:portType is an abstract construct.

819 wsdp:Profile/@wsp:Optional="true"

820 Per WS-Policy [[WS-Policy](#)], this is compact notation for two policy alternatives,
821 one with and one without the assertion. The intuition is that the behavior
822 indicated by the assertion is optional, or in this case, that the SERVICE supports
823 but does not require compliance with this profile.

824 CORRECT:

```

825 <wsp:Policy
826     xmlns:wsdp="http://schemas.xmlsoap.org/ws/2006/02/devprof"
827     xmlns:wsp="http://schemas.xmlsoap.org/ws/2004/09/policy" >
828     <wsdp:Profile />
829 </wsp:Policy>
830

```

831 **R2037: A SERVICE MUST include the wsdp:Profile assertion in its policy.**

832 This assertion has Endpoint Policy Subject: a policy expression containing this
833 assertion MAY be attached to a wsdl:port, SHOULD be attached to a wsdl:binding,
834 but MUST NOT be attached to a wsdl:portType; the latter is prohibited because this
835 assertion specifies concrete behavior whereas the wsdl:portType is an abstract
836 construct.

837 **R2041: If a SERVICE uses wsp:PolicyReference/@URI to attach a policy identified by
838 an absolute URI, the SERVICE MUST have a Metadata Section with Dialect
839 equal to "http://schemas.xmlsoap.org/ws/2004/09/policy" and Identifier
840 equal to that URI.**

841 **R2025: If a SERVICE uses wsp:PolicyReference/@URI to attach a policy identified by
842 an absolute URI, then in a Get Response SOAP ENVELOPE, the SERVICE MUST
843 include the Metadata Section with Dialect equal to**

844 *"http://schemas.xmlsoap.org/ws/2004/09/policy" and Identifier equal to that*
845 *URI.*

846 *R2035: If a SERVICE uses wsp:PolicyReference/@URI to attach a policy identified by*
847 *a relative URI, the SERVICE MUST embed that policy as a child of*
848 *wSDL:definitions, and the policy MUST have a @wsu:Id containing that URI.*

849 *R2036: A SERVICE MUST NOT use @wsp:PolicyURIs to attach policy.*

850 Because all components in WSDL are extensible via elements [[BP 1.1, Section 4](#)],
851 attachment using wsp:PolicyReference/@URI is sufficient.

852 Get [[WS-Transfer](#)] is the interoperable means for a CLIENT to retrieve attached
853 policy.

854 CORRECT:

```
855 <soap:Envelope
856   xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
857   xmlns:wSDL="http://schemas.xmlsoap.org/wSDL/"
858   xmlns:wSDP="http://schemas.xmlsoap.org/ws/2006/02/devprof"
859   xmlns:wSOAP="http://schemas.xmlsoap.org/wSDL/soap12/"
860   xmlns:wSP="http://schemas.xmlsoap.org/ws/2004/09/policy"
861   xmlns:wSU="
862 = "http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-
863 utility-1.0.xsd"
864   xmlns:wSX="http://schemas.xmlsoap.org/ws/2004/09/mex"
865   xmlns:wSA="http://schemas.xmlsoap.org/ws/2004/08/addressing" >
866 <soap:Header>
867   <wSA:Action>
868     http://schemas.xmlsoap.org/ws/2004/09/transfer/GetResponse
869   </wSA:Action>
870   <wSA:RelatesTo>
871     urn:uuid:82204a83-52f6-475c-9708-174fa27659ec
872   </wSA:RelatesTo>
873   <wSA:To>
874     http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous
875   </wSA:To>
876 </soap:Header>
877 <soap:Body>
878   <wsx:Metadata>
879     <wsx:MetadataSection
880       Dialect="http://schemas.xmlsoap.org/wSDL/" >
881       <wSDL:definitions
882         targetNamespace="http://acme.example.com/colorbeam"
883         xmlns:image="http://printer.example.org/imaging" >
884         <wSP:Policy wsu:Id="DpPolicy" >
885           <wSDP:Profile />
886         </wSP:Policy>
887
888         <!-- Other WSDL components omitted for brevity. -->
889
890         <wSDL:binding name="PrintBinding" type="image:PrintPortType" >
891           <wSP:PolicyReference URI="#DpPolicy"
892             wSDL:required="true" />
893           <!-- Other WSDL components omitted for brevity. -->
894         </wSDL:binding>
895
896   </wsx:Metadata>
```

```

897 </wsx:MetadataSection>
898
899 <!-- Other Metadata Sections omitted for brevity. -->
900
901 </wsx:Metadata>
902 </soap:Body>
903 </soap:Envelope>
904

```

905 6. Eventing

906 The scope of this section is the following set of Web services specifications. All of the
907 requirements in these specifications are included by reference except where
908 superseded by normative statements herein:

- 909 • [\[WS-Eventing\]](#)

910 6.1 Subscription

911 *R3009: A HOSTED SERVICE MUST at least support Push Delivery Mode indicated by*
912 *"http://schemas.xmlsoap.org/ws/2004/08/eventing/DeliveryModes/Push".*

913 *R3010: A HOSTED SERVICE MUST NOT generate a*
914 *wse:DeliveryModeRequestedUnavailable SOAP Fault in response to a*
915 *Subscribe SOAP ENVELOPE with a Delivery Mode of*
916 *"http://schemas.xmlsoap.org/ws/2004/08/eventing/DeliveryModes/Push".*

917 The Push Delivery Mode [\[WS-Eventing\]](#) is the default Delivery Mode and indicates
918 the Event Source (HOSTED SERVICE) will push Notifications to the Event Sink
919 (CLIENT).

920 *R3017: If a HOSTED SERVICE does not understand the [address] of the Notify To of*
921 *a Subscribe SOAP ENVELOPE, the HOSTED SERVICE MUST generate a*
922 *wsa:DestinationUnreachable SOAP Fault.*

923 *R3018: If a HOSTED SERVICE does not understand the [address] of the End To of a*
924 *Subscribe SOAP ENVELOPE, the HOSTED SERVICE MUST generate a*
925 *wsa:DestinationUnreachable SOAP Fault.*

926 *R3019: If a HOSTED SERVICE cannot deliver a Notification SOAP ENVELOPE to an*
927 *Event Sink, the HOSTED SERVICE MAY terminate the corresponding*
928 *Subscription and SHOULD send a Subscription End SOAP ENVELOPE with a*
929 *Status of*
930 *"http://schemas.xmlsoap.org/ws/2004/08/eventing/DeliveryFailure".*

931 6.1.1 Filtering

932 To enable subscribing to one or more Notifications exposed by a HOSTED SERVICE,
933 this profile defines a Filter Dialect designated
934 "http://schemas.xmlsoap.org/ws/2006/02/devprof/Action".

- 935 • A Filter in this Dialect contains a white space-delimited list of URIs that indicate
936 the **[action]** property of desired Notifications.
- 937 • The content of a Filter in this Dialect is defined as
938 `xs:list/@itemType="xs:anyURI" \[XML Schema Part 2\].`
- 939 • A Filter in this Dialect evaluates to true for an Output Message of a Notification or
940 Solicit-Response operation if and only if a URI in the Filter matches the **[action]**

941 property of the Message using the
942 "http://schemas.xmlsoap.org/ws/2005/04/discovery/rfc2396" matching rule
943 [[WS-Discovery](#)].

944 The Action Dialect uses the RFC 2396 prefix matching rule so CLIENTs can subscribe
945 to a related set of Notifications by including the common prefix of the **[action]**
946 property of those Notifications. Typically, the Notifications within a WSDL portType
947 [[WSDL 1.1](#)] will share a common **[action]** property prefix, and specifying that prefix
948 with the Action Dialect will be a convenient means to subscribe to all Notifications
949 defined by a portType.

950 *R3008: A HOSTED SERVICE MUST at least support Filtering by the Dialect*
951 *"http://schemas.xmlsoap.org/ws/2006/02/devprof/Action".*

952 CORRECT:

```
953 <soap:Envelope
954   xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
955   xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing"
956   xmlns:wse="http://schemas.xmlsoap.org/ws/2004/08/eventing" >
957   <soap:Header>
958     <wsa:Action>
959       http://schemas.xmlsoap.org/ws/2004/08/eventing/Subscribe
960     </wsa:Action>
961     <wsa:MessageID>
962       urn:uuid:314bea3b-03af-47a1-8284-f495497f1e33
963     </wsa:MessageID>
964     <wsa:ReplyTo>
965       <wsa:Address>
966         http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous
967       </wsa:Address>
968     </wsa:ReplyTo>
969     <wsa:To>http://172.30.184.244/print</wsa:To>
970   </soap:Header>
971   <soap:Body>
972     <wse:Subscribe>
973       <wse:Delivery>
974         <wse:NotifyTo>
975           <wsa:Address>
976             urn:uuid:3726983d-02de-4d41-8207-d028ae92ce3d
977           </wsa:Address>
978         </wse:NotifyTo>
979       </wse:Delivery>
980       <wse:Expires>PT10M</wse:Expires>
981       <wse:Filter
982         Dialect="http://schemas.xmlsoap.org/ws/2006/02/devprof/Action"
983       >
984         http://printer.example.org/imaging/PrintBasicPortType/JobEndState
985         http://printer.example.org/imaging/PrintBasicPortType/PrinterState
986       </wse:Filter>
987     </wse:Subscribe>
988   </soap:Body>
989 </soap:Envelope>
990
```

991 *R3011: A HOSTED SERVICE MUST NOT generate a wse:FilteringNotSupported SOAP*
992 *Fault in response to a Subscribe SOAP ENVELOPE.*

993 A HOSTED SERVICE must support filtering, at least by **[action]**, so the Filtering Not
994 Supported SOAP Fault is not appropriate.

995 *R3012: A HOSTED SERVICE MUST NOT generate a*
996 *wse:FilteringRequestedUnavailable SOAP Fault in response to a Subscribe*
997 *SOAP ENVELOPE with a Filter Dialect of*
998 *"http://schemas.xmlsoap.org/ws/2006/02/devprof/Action".*

999 To indicate that a HOSTED SERVICE does not expose any Notifications that would
1000 match the contents of a Filter with the Action Dialect, this profile defines the
1001 following SOAP Fault:

[action]	http://schemas.xmlsoap.org/ws/2006/02/devprof/Fault
[Code]	soap:Sender
[Subcode]	wsdp:FilterActionNotSupported
[Reason]	E.g., "no notifications match the supplied filter"
[Detail]	(None defined.)

1002 *R3020: If none of the Notifications exposed by a HOSTED SERVICE match the*
1003 ***[action]** values in a Subscribe SOAP ENVELOPE Filter whose Dialect is*
1004 *"http://schemas.xmlsoap.org/ws/2006/02/devprof/Action", the HOSTED*
1005 *SERVICE MUST generate a wsdp:FilterActionNotSupported SOAP Fault.*

1006 6.2 Subscription Duration and Renewal

1007 *R3005: If a Subscribe SOAP ENVELOPE contains a requested Expiration of type*
1008 *xs:dateTime, the HOSTED SERVICE MAY include an Expiration of type*
1009 *xs:duration in the Subscribe Response SOAP ENVELOPE.*

1010 *R3006: If a Renew SOAP ENVELOPE contains a requested Expiration of type*
1011 *xs:dateTime, the HOSTED SERVICE MAY include an Expiration of type*
1012 *xs:duration in the Renew Response SOAP ENVELOPE.*

1013 *R3016: A HOSTED SERVICE MUST NOT generate a wse:UnsupportedExpirationType*
1014 *SOAP Fault in response to a Subscribe or Renew SOAP ENVELOPE with an*
1015 *Expiration type of xs:duration.*

1016 *R3013: A HOSTED SERVICE MAY generate a wse:UnsupportedExpirationType SOAP*
1017 *Fault in response to a Subscribe or Renew SOAP ENVELOPE with an Expiration*
1018 *of type xs:dateTime.*

1019 Event Sources are required to have an internal clock, but there is no requirement
1020 that the clock be synchronized with other HOSTED SERVICES. Therefore, Event
1021 Sources are required to express Subscription Expiration as a duration but are not
1022 required to express Subscription Expiration as an absolute time.

1023 *R3015: A HOSTED SERVICE MAY generate a wsa:ActionNotSupported SOAP Fault in*
1024 *response to a Get Status SOAP ENVELOPE.*

1025 Event Sources are not required to support retrieving subscription status.

1026 7. Security

1027 This section defines a RECOMMENDED baseline for interoperable security between a
1028 DEVICE and a CLIENT. A DEVICE (or CLIENT) is free to support other security

1029 mechanisms in addition to, or in place of, this mechanism as specified by WSDL
1030 [[WSDL 1.1](#)], policies [[WS-Policy](#)], or other mechanisms. In the absence of an explicit
1031 indication stating that a different security mechanism is to be used, the default
1032 security mechanism defined here is assumed to apply.

1033 This section defines the protocols and message formats required to authenticate a
1034 DEVICE and securely communicate with a DEVICE. It references well-known
1035 algorithms and protocols for authentication, establishment of a session key, and
1036 encryption.

1037 This scope of this section is the following set of Web services specifications. All of the
1038 requirements in these specifications are included by reference except where
1039 superseded by normative statements herein:

- 1040 • [[AES/TLS](#)]
- 1041 • [[HTTP Authentication](#)]
- 1042 • [[SHA1](#)]
- 1043 • [[TLS](#)]
- 1044 • [[RFC 4122](#)]
- 1045 • [[X.509.v3](#)]

1046 **7.1 Secure Communication**

1047 **7.1.1 Integrity**

1048 Integrity is the process that protects MESSAGES against tampering while in transit.
1049 Integrity is an optional component of DEVICE security. However, if provided,
1050 integrity MUST adhere to the following requirements:

1051 *R4000: A SERVICE MUST not send a SOAP ENVELOPE without protecting the*
1052 *integrity of any Message Information Header blocks matching the following*
1053 *XPath expressions: (a) /soap:Envelope/soap:Header/wsa:Action, (b)*
1054 */soap:Envelope/soap:Header/wsa:MessageID, (c)*
1055 */soap:Envelope/soap:Header/wsa:To, (d)*
1056 */soap:Envelope/soap:Header/wsa:ReplyTo, (e)*
1057 */soap:Envelope/soap:Header/wsa:RelatesTo.*

1058 *R4063: A SERVICE MAY reject a SOAP ENVELOPE that has unprotected Message*
1059 *Information Header blocks.*

1060 *R4001: A SERVICE MUST not send a SOAP ENVELOPE without protecting the*
1061 *integrity of the SOAP ENVELOPE Body in conjunction with any Message*
1062 *Information Block(s) from R4000.*

1063 *R4064: A SERVICE MAY reject a SOAP ENVELOPE that does not protect the integrity*
1064 *of the SOAP ENVELOPE Body.*

1065 In this profile, the integrity of discovery SOAP ENVELOPES is protected using
1066 message-level signatures, while the integrity of other MESSAGES is protected using a
1067 Secure Channel. Other profiles may use alternate mechanisms to protect the
1068 integrity of MESSAGES.

1069 **7.1.2 Confidentiality**

1070 Confidentiality is the process by which sensitive information is protected against
1071 unauthorized disclosure. Confidentiality is an optional component of DEVICE security;
1072 however, if provided, confidentiality MUST adhere to the following requirements:

1073 *R4002: A SERVICE MUST NOT send a SOAP ENVELOPE without encrypting the SOAP*
1074 *ENVELOPE Body.*

1075 *R4067: A SERVICE MAY reject a SOAP ENVELOPE that does not encrypt the SOAP*
1076 *ENVELOPE Body.*

1077 *R4003: A SENDER MUST provide key transfer information to authorized RECEIVERS.*

1078 In this profile, discovery MESSAGES are not encrypted, while other MESSAGES are
1079 encrypted using a Secure Channel. Other profiles may use alternate mechanisms to
1080 encrypt MESSAGES.

1081 **7.1.3 Authentication**

1082 Authentication is the process by which the identity of the sender is determined by
1083 the recipient. Authentication is an optional component of DEVICE security; however,
1084 if provided, authentication MUST adhere to the following requirements:

1085 *R4004: A SENDER MUST authenticate itself to a RECEIVER using credentials*
1086 *acceptable to the RECEIVER.*

1087 In this profile, authentication is done using certificates, either through a shared trust
1088 root or through a PIN / Password exchanged out of band. Other profiles may use
1089 alternate authentication mechanisms.

1090 If multicast messages are secured, the following additional requirements apply:

1091 *R4005: On multicast MESSAGES, a CLIENT MUST use an authentication credential*
1092 *that is suitable for all DEVICES that could legitimately process the multicast*
1093 *MESSAGE.*

1094 **7.1.4 Trust**

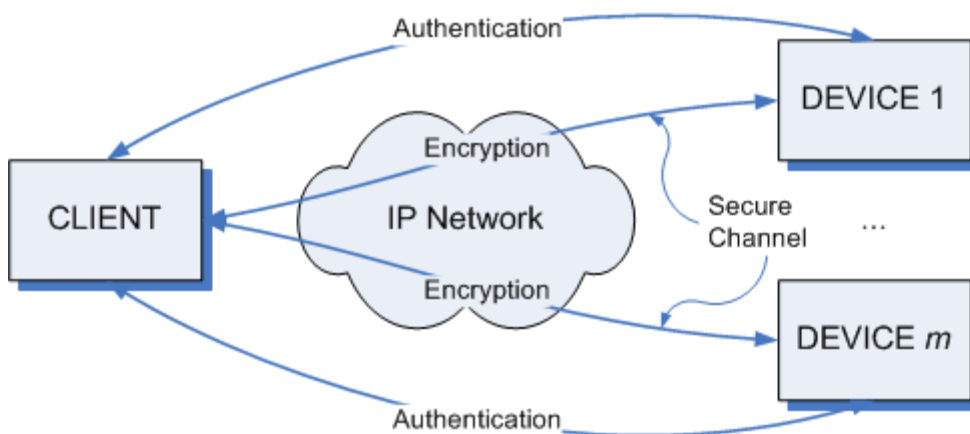
1095 There are different trust models associated with DEVICE security. The following
1096 requirements profile the kinds of trust that may be used with DEVICE security in this
1097 profile.

1098 *R4007: CLIENTs and DEVICEs MUST have the necessary credentials to perform*
1099 *authentication.*

1100 The distribution of the credentials needed for establishing the trust relationship is out
1101 of the scope of this profile. The level of security as well as the supported protocols
1102 for a given CLIENT - DEVICE relationship are advertised in the policy assertions of
1103 the discovery MESSAGES defined herein.

1104 *R4008: A SERVICE MAY use additional mechanisms to verify the authenticity of the*
1105 *SENDER of any received MESSAGE by analyzing information provided by the*
1106 *lower networking layers.*

1107 **7.1.5 Network Model**



1108

1109 Following authentication, a DEVICE and a CLIENT communicate over a Secure (i.e.,
1110 encrypted) Channel. The network is an IP-based network that can span one or more
1111 administrative domains (such as a workgroup subnet), a domain comprised of
1112 multiple subnets, or comprised of multiple administrative domains (such as the
1113 global Internet). The level of security is determined by the security policies of the
1114 administrative domain, which may vary between different environments.

1115 **R4009: Security MUST be applied for all MESSAGES received from, sent to, or**
1116 **traversed through other administrative domains.**

1117 It is assumed that MESSAGES received from/via other administrative domains cannot
1118 be trusted.

1119 **R4010: Except for MESSAGES exchanged during discovery, security SHALL be applied**
1120 **at the Transport level. Discovery relies on MESSAGE security.**

1121 **7.1.6 Security Association**

1122 DEVICE association encompasses mutual authentication of DEVICE and CLIENT as
1123 well as the establishment of a Secure Transport Channel over which the subsequent
1124 communication between the CLIENT and the DEVICE takes place. The CLIENT
1125 security requirements are advertised by the CLIENT during discovery as part of the
1126 policy assertions carried in the respective Probe and Resolve SOAP ENVELOPES.
1127 Security requirements can range from no security required to authentication and
1128 communication over a Secure (i.e., encrypted) Channel.

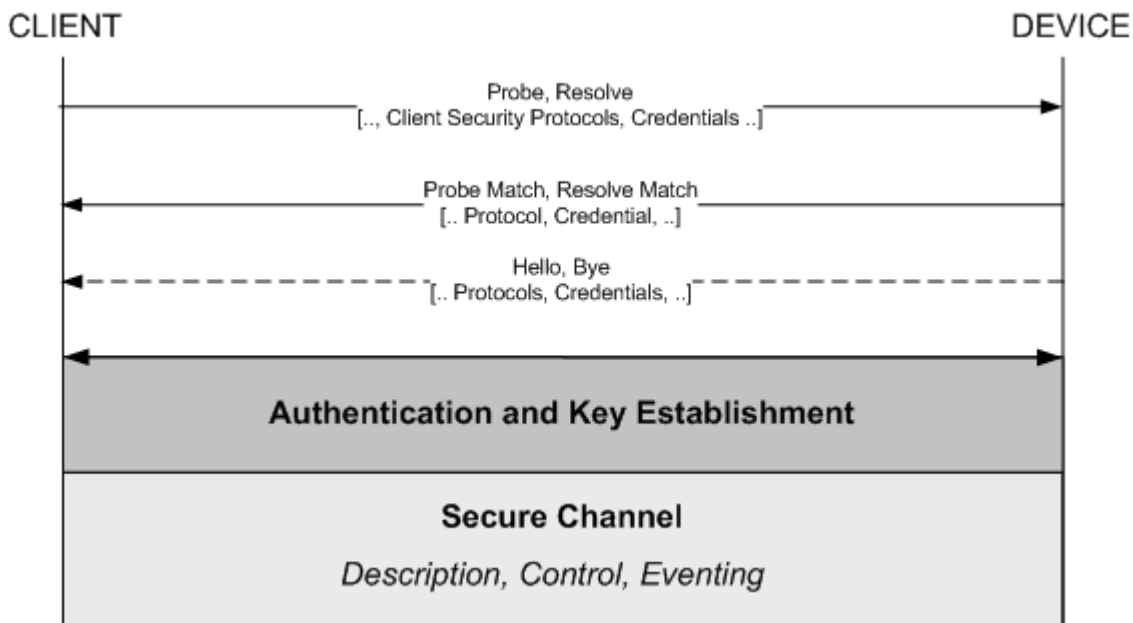
1129 The supported protocols for authentication and key establishment are advertised and
1130 negotiated during discovery.

1131 **R4068: The CLIENT MAY include policy assertions in the Probe and Resolve SOAP**
1132 **ENVELOPES containing the protocols it supports. If the CLIENT includes**
1133 **multiple protocols, the protocols MUST be ordered with decreasing**
1134 **preference, i.e., the first protocol listed is the preferred protocol the client**
1135 **wishes to use.**

1136 **R4012: The DEVICE MUST select the protocol from the list of received protocols it**
1137 **wishes to use for authentication and key establishment, and the DEVICE**
1138 **MUST include the selected protocol in the policy assertion of the respective**
1139 **Probe Match or Resolve Match SOAP ENVELOPE.**

1140 *R4013: Following discovery, the CLIENT MUST invoke the association process by*
 1141 *authenticating the DEVICE using a protocol for security and parameters*
 1142 *supported by both CLIENT and DEVICE as negotiated via Policy for the EPR.*

1143 The sequence for authentication and establishment of a Secure Channel is illustrated
 1144 below. It is assumed that credentials (certificates, shared secrets) are established by
 1145 an out-of-band mechanism prior or during the association phase. The out-of-band
 1146 mechanism is out of the scope of this profile. If the authentication is successful, a
 1147 Secure Channel is established. Subsequent operations like description, control, and
 1148 eventing use the Secure Channel.



1149
 1150 Once the DEVICE leaves the network, i.e., the DEVICE sends a Bye SOAP ENVELOPE,
 1151 the Secure Channel is removed, and the authentication information as well as
 1152 session keys become invalid.

1153 7.1.7 DEVICE Behavior

1154 *R4014: A DEVICE MAY require authentication of a CLIENT.*

1155 *R4015: To verify the authenticity of multicast messages sent by the DEVICE during*
 1156 *discovery, i.e., Hello and Bye SOAP ENVELOPES, multicast MESSAGES*
 1157 *SHOULD be signed.*

1158 *R4016: Unicast MESSAGES sent by a DEVICE in response to multicast MESSAGES,*
 1159 *i.e., Probe Match and Resolve Match SOAP ENVELOPES, SHOULD be signed.*

1160 *R4017: A CLIENT MAY ignore MESSAGES received during discovery that have no*
 1161 *signature or a nonverifiable signature.*

1162 *R4018: A DEVICE SHOULD cache authentication information for a CLIENT as valid as*
 1163 *long as the DEVICE is connected to the CLIENT.*

1164 7.1.8 Security Protocols and Credentials

1165 *R4025: A CLIENT MUST indicate the Security protocols and Credentials for*
 1166 *authentication and key establishment it supports in /soap:Envelope/*

1167 *soap:Header/ wsa:ReplyTo/ wsx:Metadata of a Probe and/or Resolve SOAP*
1168 *ENVELOPE.*

1169 *R4026: A DEVICE SHALL select from the list of Security Protocols and Credentials*
1170 *indicated by the CLIENT which Security Protocol the DEVICE wishes to use*
1171 *and return that selection in /soap:Envelope/ soap:Body/ */*
1172 *wsa:EndpointReference/ wsx:Metadata of the corresponding Probe Match (or*
1173 *Resolve Match) SOAP ENVELOPE.*

1174 Embedding a Metadata element [[WS-MetadataExchange](#)] within the extension point
1175 of an Endpoint Reference [[WS-Addressing](#)] is a means to provide metadata about the
1176 endpoint. This use of the Metadata element generalizes the existing **[policy]**
1177 property [[WS-Addressing](#)] and is the expected means to express WS-Policy in future
1178 versions of WS-Addressing.

1179 *R4027: A CLIENT MUST use the Security Protocol and Credential indicated by the*
1180 *DEVICE in the Probe Match (or Resolve Match) SOAP ENVELOPE for*
1181 *authentication and key establishment.*

1182 *R4028: CLIENTs and DEVICEs SHOULD support the following Security Protocols and*
1183 *Credentials for authentication and key establishment: TLS with client*
1184 *certificates and server certificates, respectively.*

1185 *R4069: CLIENTs and DEVICEs MUST support HTTP Basic Authentication.*

1186 **7.1.9 Security for Discovery**

1187 In the discovery phase, the client learns of the existence of the device on the
1188 network. Subsequently, the identity of the device is verified, and the device is
1189 connected to the client. The policy assertions carried in the messages exchanged
1190 during Discovery contain the CLIENT Security Requirements as well as the Security
1191 Protocols supported by CLIENT and DEVICE for authentication and establishment of a
1192 Secure Channel.

1193 *R4029: If a DEVICE cannot meet the CLIENT Security Requirements or if a CLIENT*
1194 *and a DEVICE do not support intersecting Security Protocols and Credentials,*
1195 *no association SHALL take place.*

1196 Probe

1197 A CLIENT initiates the discovery process by probing the network for a DEVICE it is
1198 interested in.

1199 *R4030: A Probe SOAP ENVELOPE SHOULD contain the Security Protocols and*
1200 *Credentials in /soap:Envelope/ soap:Header/ wsa:ReplyTo/ wsp:Policy.*

1201 *R4031: In the absence of any policy assertion for security, no security SHALL be*
1202 *required.*

1203 *R4032: A Device MUST NOT send a Probe Match SOAP ENVELOPE if any of the*
1204 *following are true: (a) the DEVICE is outside the local subnet of the CLIENT,*
1205 *and the Probe SOAP ENVELOPE was sent as multicast, or (b) the DEVICE does*
1206 *not support the indicated CLIENT Security Protocols and Credentials.*

1207 *R4065: A CLIENT MUST discard a Probe Match SOAP ENVELOPE if it is received*
1208 *MATCH_TIMEOUT seconds or more later than the last corresponding Probe*
1209 *SOAP ENVELOPE was sent.*

1210 Hello

1211 *R4034: A DEVICE SHOULD sign a Hello SOAP ENVELOPE.*

1212 One or more CLIENTs may respond to the Hello SOAP ENVELOPE and associate with
1213 the DEVICE.

1214 *R4035: If a DEVICE has multiple credentials, it SHOULD send separate Hello SOAP*
1215 *ENVELOPEs using different credentials to sign each.*

1216 Resolve

1217 *R4036: A Device MUST NOT send a Resolve Match SOAP ENVELOPE if any of the*
1218 *following are true: (a) the DEVICE is outside the local subnet of the CLIENT,*
1219 *and the Probe SOAP ENVELOPE was sent as multicast, or (b) the DEVICE does*
1220 *not support the indicated CLIENT Security Protocols and Credentials.*

1221 *R4066: A CLIENT MUST discard a Resolve Match SOAP ENVELOPE if it is received*
1222 *MATCH_TIMEOUT seconds or more later than the last corresponding Resolve*
1223 *SOAP ENVELOPE was sent.*

1224 Bye

1225 *R4037: A DEVICE SHOULD sign a Bye SOAP ENVELOPE.*

1226 *R4038: If a DEVICE has different credentials applicable to multiple CLIENTs, it*
1227 *SHOULD send separate Bye SOAP ENVELOPEs with the credentials for each of*
1228 *the previously associated CLIENTs.*

1229 **7.1.10 Authentication**

1230 The authentication step that follows discovery verifies the credentials of the DEVICE
1231 and CLIENT in a secure manner. In addition to verifying the credentials, a session
1232 key is established in the authentication handshake. Credentials may be cached on
1233 the DEVICE and/or CLIENT to simplify subsequent authentications. The CLIENT
1234 invokes the authentication process using the protocols and credentials indicated in
1235 the DEVICE policy assertions conveyed during the discovery phase.

1236 Transport Layer Security (TLS)

1237 TLS provides mutual authentication of CLIENT and DEVICE as well as the
1238 establishment of a Secure Channel over which MESSAGEs are exchanged in a secure
1239 manner.

1240 DEVICE Authentication with TLS

1241 *R4039: If TLS is negotiated as the Security Protocol, the CLIENT MUST initiate*
1242 *authentication with the DEVICE by setting up a TLS session.*

1243 *R4070: A DEVICE MUST indicate the use of TLS for a MESSAGE exchange using the*
1244 *"https" scheme URI contained in the DEVICE description and WSDL.*

1245 *R4042: Following the establishment of a Secure Channel using TLS, subsequent*
1246 *MESSAGE exchanges over HTTP SHOULD use an existing TLS session.*

1247 Certificates

1248 *R4043: Each DEVICE SHOULD have its own, unique Certificate.*

1249 The Certificate contains information pertinent to the specific device including its
1250 public key. Typically, certificates are issued by a trusted authority or a delegate (2nd
1251 tier) or a delegate of the delegate.

1252 *R4045: The format of the certificate MUST follow the common standard X.509v3.*

1253 An example of a self-signed X.509 certificate is shown below.

Type	Element	Usage	Example
Type	Element	Usage	Example
Basic Elements	Version	TLS	3
	Certificate Serial Number		1234567
	Signature Algorithm Identifier		RSA
	Issuer		a7731471-4b54-4a64-942c-7d481dcb9614
	Validity Period		11/09/2001 - 01/07/2015
	Subject	UUID	a7731471-4b54-4a64-942c-7d481dcb9614
	Subject Public Key Information		rsaEncryption 1024 10888232e76740bd873462ea2c64ca1d a6f9112656a34b949d32cede0e476547 84ba0f7e62e143429d3217ee45ce5304 308e65a6eee6474cb4d9a3c0295c8267 761661ccb7546a09d5f03a8ea3b1160 dac9fb6e6ba94e54b6c8ee892e492f4c e3a96bbd9d7b4c4bb98b7c052ff361ba cee01718122c4f0d826efc123bb1b03d
Extensions	Extended Key Usage	Server Authentication	1.3.6.1.5.5.7.3.1
		Client Authentication	1.3.6.1.5.5.7.3.2
Signature	Certification Authority's Digital Signature		5938f9908916cca32321916a184a6e75 2becb14fb99c4f33a03b03c3c752117c 91b8fb163d3541fca78bca235908ba69 1f7e36004a2d499a8e23951bd8af961d 36be05307ec34467a7c66fbb7fb5e49c 25e8dbdae4084ca9ba244b5bc1a377e5 262b9ef543ce47ad8a6b1d28c9138d0a dc8f5e3b469e42a5842221f9cf0a50d1

1254 The Subject field (listed above) contains the UUID in string representation format.

1255 Certificate management is out of the scope of this profile.

1256 TLS Authentication with Client Certificate

1257 *R4071: If the CLIENT and the DEVICE exchanged certificates during the TLS*
1258 *handshake, and the DEVICE as well as the CLIENT were able to verify the*
1259 *certificates, the CLIENT and DEVICE are mutually authenticated, and no*
1260 *further steps SHALL be required.*

1261 *R4046: A DEVICE MAY require an additional authentication step after the TLS*
1262 *handshake, if the DEVICE was not able to verify the certificate, or if the*
1263 *CLIENT did not provide a certificate during the TLS handshake.*

1264 *R4047: A DEVICE MAY require HTTP Authentication.*

1265 *R4048: If the HTTP authentication is successful, and the CLIENT presents a*
1266 *certificate to the DEVICE, the DEVICE SHOULD cache the certificate in its local*
1267 *certificate store of trusted certificates for future authentication of the CLIENT.*

1268 This avoids the need for HTTP authentication for subsequent associations.

1269 HTTP Authentication

1270 *R4049: The CLIENT MAY be required to authenticate itself to the DEVICE during the*
1271 *association phase.*

1272 HTTP authentication requires credentials in the form of username and password. It is
1273 assumed that how the CLIENT and DEVICE share knowledge of the username and
1274 password is out-of-band and beyond the scope of this profile.

1275 Because the authentication is performed over the Secure Channel established during
1276 TLS handshake, HTTP Basic authentication may be used safely.

1277 *R4050: If a DEVICE requires HTTP authentication, the DEVICE SHALL challenge the*
1278 *CLIENT using the HTTP 401 response code.*

1279 *R4051: A CLIENT MUST authenticate using one of the options listed in the HTTP-*
1280 *Authenticate header.*

1281 *R4052: HTTP Authentication MUST use the following parameters for username and*
1282 *password of the HTTP Request: UserName, PIN / Password.*

1283 The UserName is supplied to the DEVICE during HTTP authentication and MAY be
1284 used for establishing multiple access control classes, such as administrators, users,
1285 and guests. The naming and use of UserName is implementation-dependent and out
1286 of the scope of this profile.

1287 *R4053: If no UserName is provided, "admin" SHALL be used as the default*
1288 *UserName.*

1289 The purpose of the PIN / Password is to authenticate the CLIENT to the DEVICE
1290 during the HTTP authentication. In addition, the PIN / Password verifies the
1291 certificate that the DEVICE supplied during the TLS handshake.

1292 *R4054: The RECOMMENDED size of a PIN / Password is at least 8 characters using at*
1293 *least a 32 character alphabet.*

1294 *R4055: The PIN / Password that is unique to the DEVICE SHALL be conveyed to the*
1295 *CLIENT out-of-band. The methods of conveying the PIN out-of-band are out*
1296 *of the scope of this profile.*

1297 *R4056: To reduce the attack surface, the DEVICE and CLIENT MAY limit the number*
1298 *of failed authentication attempts as well as the time interval successive*
1299 *attempts are made for one TLS session.*

1300 Upon successful authentication, the DEVICE is associated with the CLIENT.

1301 **7.1.11 Secure Channel**

1302 Following Authentication, a Secure (i.e., encrypted) Channel at the transport level is
1303 established between CLIENT and DEVICE.

1304 *R4057: All secure communication for Description, Control, and Eventing between the*
1305 *CLIENT and DEVICE MUST use the Secure Channel. The protocols for*
1306 *encryption as well as the keys used for encryption are negotiated during the*
1307 *authentication phase.*

1308 *R4072: A DEVICE MUST support receiving and responding to a Probe SOAP*
1309 *ENVELOPE over HTTP using the Secure Channel.*

1310 *R4073: A DEVICE MAY ignore a Probe SOAP ENVELOPE sent over HTTP that does not*
1311 *use the Secure Channel.*

1312 As prescribed by R1015, a CLIENT may send a Probe over HTTP; this Probe (and
1313 Probe Match, if any) are sent using the Secure Channel.

1314 **7.1.12 TLS Ciphersuites**

1315 *R4059: It is the responsibility of the sender to convert the embedded URL to use*
1316 *HTTPS as different transport security mechanisms can be negotiated.*

1317 *R4060: A DEVICE MUST support the following TLS Ciphersuite:*
1318 *TLS_RSA_WITH_RC4_128_SHA.*

1319 *R4061: It is recommended that a DEVICE also support the following TLS Ciphersuite:*
1320 *TLS_RSA_WITH_AES_128_CBC_SHA.*

1321 *R4062: Additional Ciphersuites MAY be supported. They are negotiated during the*
1322 *TLS handshake.*

1323 **8. Acknowledgements**

1324 This profile has been developed as a result of joint work with many individuals and
1325 teams, including: Don Box (Microsoft), Dan Driscoll (Microsoft), Mike Fenelon
1326 (Microsoft), Omri Gazitt (Microsoft), Bertus Greeff (Microsoft), Rob Hain (Microsoft),
1327 Rich Hasha (Microsoft), Gopal Kakivaya (Microsoft), Chris Kurt (Microsoft), David
1328 Lindsey (Lexmark), Jonathan Marsh (Microsoft), Henry Rawas (Microsoft), Sam
1329 Rhodus (Lexmark), Adam Sapek (Microsoft), Stacy Simpson (Lexmark), Lifen Tian
1330 (Ricoh), David Turner (Microsoft), Mike Vernal (Microsoft), Yaotian Wang (Ricoh),
1331 Kenny Wolf (Microsoft).

1332 **9. References**

1333 **[AES/TLS]**

1334 P. Chown, "Advanced Encryption Standard (AES) Ciphersuites for Transport Layer
1335 Security (TLS)," June 2004. (See <http://www.ietf.org/rfc/rfc3268.txt>)

1336 **[BP 1.1, Section 4]**

1337 K. Ballinger, et al, "Basic Profile Version 1.1, Section 4: Service Description,"
1338 August 2004. (See [http://www.ws-i.org/Profiles/BasicProfile-1.1-2004-08-
1339 24.html#description](http://www.ws-i.org/Profiles/BasicProfile-1.1-2004-08-24.html#description))

1340 **[HTTP/1.1]**

1341 R. Fielding, et al, "Hypertext Transfer Protocol -- HTTP/1.1," June 1999. (See
1342 <http://www.ietf.org/rfc/rfc2616.txt>)

1343 **[HTTP Authentication]**
1344 J. Franks, et al, "HTTP Authentication: Basic and Digest Access Authentication,"
1345 June 1999. (See <http://www.ietf.org/rfc/rfc2617.txt>)

1346 **[MIME]**
1347 N. Freed, et al, "Multipurpose Internet Mail Extensions (MIME) Part One: Format
1348 of Internet Message Bodies," November 1996. (See
1349 <http://www.ietf.org/rfc/rfc2045.txt>)

1350 **[MTOM]**
1351 N. Mendelsohn, et al, "SOAP Message Transmission Optimization Mechanism,"
1352 January 2005. (See <http://www.w3.org/TR/2005/REC-soap12-mtom-20050125/>
1353)

1354 **[RFC 4122]**
1355 P. Leach, et al, "A Universally Unique Identifier (UUID) URN Namespace," July
1356 2005. (See <http://www.ietf.org/rfc/rfc4122.txt>)

1357 **[SHA1]**
1358 "Secure Hash Standard," April 1995. (See
1359 <http://www.itl.nist.gov/fipspubs/fip180-1.htm>)

1360 **[SOAP 1.2, Part 1]**
1361 M. Gudgin, et al, "SOAP Version 1.2 Part 1: Messaging Framework," June 2003.
1362 (See <http://www.w3.org/TR/2003/REC-soap12-part1-20030624/>)

1363 **[SOAP 1.2, Part 2, Section 7]**
1364 M. Gudgin, et al, " SOAP Version 1.2 Part 2: Adjuncts, Section 7: SOAP HTTP
1365 Binding," June 2003. (See [http://www.w3.org/TR/2003/REC-soap12-part2-
1366 20030624/#soapinhttp](http://www.w3.org/TR/2003/REC-soap12-part2-20030624/#soapinhttp))

1367 **[SOAP-over-UDP]**
1368 H. Combs, et al, "SOAP-over-UDP," September 2004. (See
1369 <http://schemas.xmlsoap.org/ws/2004/09/soap-over-udp>)

1370 **[TLS]**
1371 T. Dierks, et al, "The TLS Protocol, Version 1.0," January 1999. (See
1372 <http://www.ietf.org/rfc/rfc2246.txt>)

1373 **[WS-Addressing]**
1374 D. Box, et al, "Web Services Addressing (WS-Addressing)," August 2004. (See
1375 <http://www.w3.org/Submission/2004/SUBM-ws-addressing-20040810/>)

1376 **[WS-Discovery]**
1377 J. Beatty, et al, "Web Services Dynamic Discovery (WS-Discovery)," April 2005.
1378 (See <http://schemas.xmlsoap.org/ws/2005/04/discovery>)

1379 **[WSDL 1.1]**
1380 E. Christensen, et al, "Web Services Description Language (WSDL) 1.1," March
1381 2001. (See <http://www.w3.org/TR/2001/NOTE-wsdl-20010315>)

1382 **[WSDL Binding for SOAP 1.2]**
1383 K. Ballinger, et al, "WSDL Binding for SOAP 1.2," April 2002. (See
1384 <http://schemas.xmlsoap.org/wsdl/soap12/>)

1385 **[WS-Eventing]**

- 1386 L. Cabrera, et al, "Web Services Eventing (WS-Eventing)," August 2004. (See
1387 <http://schemas.xmlsoap.org/ws/2004/08/eventing/>)
- 1388 **[WS-MetadataExchange]**
- 1389 K. Ballinger, et al, "Web Services Metadata Exchange (WSMetadataExchange),"
1390 September 2004. (See <http://schemas.xmlsoap.org/ws/2004/09/mex/>)
- 1391 **[WS-Policy]**
- 1392 S. Bajaj, et al, "Web Services Policy Framework (WS-Policy)," September 2004.
1393 (See <http://schemas.xmlsoap.org/ws/2004/09/policy>)
- 1394 **[WS-PolicyAttachment]**
- 1395 S. Bajaj, et al, "Web Services Policy Attachment (WS-PolicyAttachment),"
1396 September 2004. (See <http://schemas.xmlsoap.org/ws/2004/09/policy>)
- 1397 **[WS-Security 2004]**
- 1398 A. Nadalin, et al, "Web Services Security: SOAP Message Security 1.0 (WS-
1399 Security 2004)," March 2004. (See [http://docs.oasis-
1400 open.org/wss/2004/01/oasis-200401-wss-soap-message-security-1.0.pdf](http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-soap-message-security-1.0.pdf))
- 1401 **[WS-Transfer 2004]**
- 1402 J.Alexander, et al, " Web Service Transfer (WS-Transfer)", September 2004. (See
1403 <http://schemas.xmlsoap.org/ws/2004/09/transfer/>)
- 1404 **[X.509.v3]**
- 1405 "ITU-T X.509.v3 Information technology - Open Systems Interconnection - The
1406 Directory: Public-key and attribute certificate frameworks (ISO/IEC/ITU 9594-
1407 8)."
- 1408 **[XML Schema, Part 1]**
- 1409 H. Thompson, et al, "XML Schema Part 1: Structures," May 2001. (See
1410 <http://www.w3.org/TR/2001/REC-xmlschema-1-20010502/>)
- 1411 **[XML Schema, Part 2]**
- 1412 P. Biron, et al, "XML Schema Part 2: Datatypes," May 2001. (See
1413 <http://www.w3.org/TR/2001/REC-xmlschema-2-20010502/>)
- 1414 **10. Informative References**
- 1415 The following documents are referenced for informational purposes only. They are
1416 not part of the scope of the profile:
- 1417 **[IPv6 Autoconfig]**
- 1418 S. Thomson, et al, "IPv6 Stateless Address Autoconfiguration," December 1998.
1419 (See <http://www.ietf.org/rfc/rfc2462.txt>)
- 1420 **[DHCP]**
- 1421 R. Droms, "Dynamic Host Configuration Protocol," March 1997. (See
1422 <http://www.ietf.org/rfc/rfc2131.txt>)
- 1423 **[RFC 2119]**
- 1424 S. Bradner, "Key words for use in RFCs to Indicate Requirement Levels," RFC
1425 2119, March 1997. (See <http://www.ietf.org/rfc/rfc2119.txt>)
- 1426 **[XML Infoset]**
- 1427 J. Cowan, et al, "XML Information Set (Second Edition)," February 2004. (See
1428 <http://www.w3.org/TR/2004/REC-xml-infoset-20040204/>)

1429 Appendix I – Constants

1430 The following constants are used throughout this profile. The values listed below
1431 supersede other values defined in other specifications listed below.

Constant	Value	Specification
APP_MAX_DELAY	5,000 milliseconds	[WS-Discovery]
DISCOVERY_PORT	3702	[WS-Discovery]
MATCH_TIMEOUT	10 seconds	[WS-Discovery]
MAX_ENVELOPE_SIZE	32,767 octets	This profile
MAX_FIELD_SIZE	256 Unicode characters	This profile
MAX_URI_SIZE	2,048 octets	This profile
MULTICAST_UDP_REPEAT	2	[SOAP-over-UDP]
UDP_MAX_DELAY	250 milliseconds	[SOAP-over-UDP]
UDP_MIN_DELAY	50 milliseconds	[SOAP-over-UDP]
UDP_UPPER_DELAY	450 milliseconds	[SOAP-over-UDP]
UNICAST_UDP_REPEAT	2	[SOAP-over-UDP]

1432 Appendix II – XML Schema

1433 A normative copy of the XML Schema [[XML Schema Part 1, Part 2](#)] description for
1434 this specification can be retrieved from the following address:

1435 <http://schemas.xmlsoap.org/ws/2006/02/devprof/devicesprofile.xsd>

1436 A non-normative copy of the XML Schema description is listed below for convenience.

```
1437 <xs:schema
1438     targetNamespace="http://schemas.xmlsoap.org/ws/2006/02/devprof"
1439     xmlns:tns="http://schemas.xmlsoap.org/ws/2006/02/devprof"
1440     xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing"
1441     xmlns:xs="http://www.w3.org/2001/XMLSchema"
1442     elementFormDefault="qualified"
1443     blockDefault="#all" >
1444
1445     <xs:import
1446         namespace="http://schemas.xmlsoap.org/ws/2004/08/addressing"
1447         schemaLocation
1448         ="http://schemas.xmlsoap.org/ws/2004/08/addressing/addressing.xsd"
1449         />
1450
1451     <xs:element name="ThisModel" type="tns:ThisModelType" />
1452     <xs:complexType name="ThisModelType" >
1453         <xs:sequence>
1454             <xs:element name="Manufacturer" type="tns:LocalizedStringType"
1455                 maxOccurs="unbounded" />
1456             <xs:element name="ManufacturerUrl" type="xs:anyURI"
1457                 minOccurs="0" />
1458             <xs:element name="ModelName" type="tns:LocalizedStringType"
1459                 maxOccurs="unbounded" />
1460             <xs:element name="ModelNumber" type="xs:string" minOccurs="0" />
```

```

1461     <xs:element name="ModelUrl" type="xs:anyURI" minOccurs="0" />
1462     <xs:element name="PresentationUrl" type="xs:anyURI"
1463         minOccurs="0" />
1464     <xs:any namespace="##other" processContents="lax"
1465         minOccurs="0" maxOccurs="unbounded" />
1466 </xs:sequence>
1467     <xs:anyAttribute namespace="##other" processContents="lax" />
1468 </xs:complexType>
1469
1470 <xs:element name="ThisDevice" type="tns:ThisDeviceType" />
1471 <xs:complexType name="ThisDeviceType" >
1472     <xs:sequence>
1473         <xs:element name="FriendlyName" type="tns:LocalizedStringType"
1474             maxOccurs="unbounded" />
1475         <xs:element name="FirmwareVersion" type="xs:string"
1476             minOccurs="0" />
1477         <xs:element name="SerialNumber" type="xs:string" minOccurs="0" />
1478         <xs:any namespace="##other" processContents="lax"
1479             minOccurs="0" maxOccurs="unbounded" />
1480     </xs:sequence>
1481     <xs:anyAttribute namespace="##other" processContents="lax" />
1482 </xs:complexType>
1483
1484 <xs:complexType name="LocalizedStringType" >
1485     <xs:simpleContent>
1486         <xs:extension base="xs:string" >
1487             <xs:anyAttribute namespace="##other" processContents="lax" />
1488         </xs:extension>
1489     </xs:simpleContent>
1490 </xs:complexType>
1491
1492 <xs:element name="Relationship" >
1493     <xs:complexType>
1494         <xs:sequence>
1495             <xs:any namespace="##any" processContents="lax"
1496                 minOccurs="0" maxOccurs="unbounded" />
1497         </xs:sequence>
1498         <xs:attribute name="Type" type="tns:DeviceRelationshipTypes"
1499             use="required" />
1500         <xs:anyAttribute namespace="##other" processContents="lax" />
1501     </xs:complexType>
1502 </xs:element>
1503
1504 <xs:simpleType name="DeviceRelationshipTypes" >
1505     <xs:union memberTypes="tns:DeviceRelationshipTypeURIs xs:anyURI" />
1506 </xs:simpleType>
1507
1508 <xs:simpleType name="DeviceRelationshipTypeURIs" >
1509     <xs:restriction base="xs:anyURI" >
1510         <xs:enumeration
1511             value="http://schemas.xmlsoap.org/ws/2006/02/devprof/host" />
1512     </xs:restriction>
1513 </xs:simpleType>
1514
1515 <xs:simpleType name="DeviceMetadataDialectURIs" >
1516     <xs:restriction base="xs:anyURI" >

```

```

1517     <xs:enumeration
1518 value="http://schemas.xmlsoap.org/ws/2006/02/devprof/ThisModel" />
1519     <xs:enumeration
1520 value="http://schemas.xmlsoap.org/ws/2006/02/devprof/ThisDevice" />
1521     <xs:enumeration
1522 value="http://schemas.xmlsoap.org/ws/2006/02/devprof/Relationship" />
1523     </xs:restriction>
1524 </xs:simpleType>
1525
1526 <xs:simpleType name="DeviceEventingFilterDialects" >
1527   <xs:union memberTypes="tns:DeviceEventingFilterDialectURIs
1528 xs:anyURI" />
1529 </xs:simpleType>
1530
1531 <xs:simpleType name="DeviceEventingFilterDialectURIs" >
1532   <xs:restriction base="xs:anyURI" >
1533     <xs:enumeration
1534 value="http://schemas.xmlsoap.org/ws/2006/02/devprof/Action" />
1535     </xs:restriction>
1536 </xs:simpleType>
1537
1538 <xs:simpleType name="DeviceActionURIs" >
1539   <xs:restriction base="xs:anyURI" >
1540     <xs:enumeration
1541 value="http://schemas.xmlsoap.org/ws/2006/02/devprof/Fault" />
1542     </xs:restriction>
1543 </xs:simpleType>
1544
1545 <xs:simpleType name="DeviceSoapFaultSubcodes" >
1546   <xs:union memberTypes="tns:DeviceSoapFaultSubcodeQNames
1547 wsa:FaultSubcodeValues xs:QName" />
1548 </xs:simpleType>
1549
1550 <xs:simpleType name="DeviceSoapFaultSubcodeQNames" >
1551   <xs:restriction base="xs:QName" >
1552     <xs:enumeration value="tns:FilterActionNotSupported" />
1553     </xs:restriction>
1554 </xs:simpleType>
1555
1556 <xs:element name="Host" type="tns:HostServiceType" />
1557 <xs:element name="Hosted" type="tns:HostServiceType" />
1558 <xs:complexType name="HostServiceType" >
1559   <xs:sequence>
1560     <xs:element ref="wsa:EndpointReference"
1561       maxOccurs="unbounded" />
1562     <xs:element ref="tns:Types" minOccurs="0" />
1563     <xs:element ref="tns:ServiceId" />
1564     <xs:any namespace="##other" processContents="lax"
1565       minOccurs="0" maxOccurs="unbounded" />
1566   </xs:sequence>
1567   <xs:anyAttribute namespace="##other" processContents="lax" />
1568 </xs:complexType>
1569
1570 <xs:element name="ServiceId" type="xs:anyURI" />
1571 <xs:element name="Types" type="tns:QNameListType" />
1572 <xs:simpleType name="QNameListType" >
1573   <xs:list itemType="xs:QName" />

```

```
1574 </xs:simpleType>
1575
1576 <xs:simpleType name="DiscoveryTypeValues" >
1577   <xs:restriction base="xs:QName" >
1578     <xs:enumeration value="tns:Device" />
1579   </xs:restriction>
1580 </xs:simpleType>
1581
1582 <xs:element name="Profile" type="tns:AssertionType" />
1583
1584 <xs:complexType name="AssertionType" >
1585   <xs:complexContent>
1586     <xs:restriction base="xs:anyType">
1587       <xs:anyAttribute namespace="##other" processContents="lax" />
1588     </xs:restriction>
1589   </xs:complexContent>
1590 </xs:complexType>
1591
1592 </xs:schema>
1593
1594
```