Devices Profile for Web Services

May 2005

Co-Developers

Shannon Chan, Microsoft
Chris Kaler, Microsoft
Thomas Kuehnel, Microsoft
Alain Regnier, Ricoh
Bryan Roe, Intel
Dale Sather, Microsoft
Jeffrey Schlimmer, Microsoft (Editor)
Hitoshi Sekine, Ricoh
Doug Walter, Microsoft
Jack Weast, Intel
Dave Whitehead, Lexmark
Don Wright, Lexmark
Yevgeniy Yarmosh, Intel

Copyright Notice

Abstract

This profile defines a minimal set of implementation constraints to enable secure Web service messaging, discovery, description, and eventing on resource-constrained endpoints.

Status

This is a public consultation draft release of this specification for community evaluation and review. We welcome feedback on this specification through the WS-* Workshop process.

Table of Contents

1. Introduction

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

- Sending secure messages to and from a Web service
- Dynamically discovering a Web service
- Describing a Web service
- Subscribing to, and receiving events from, a Web service

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

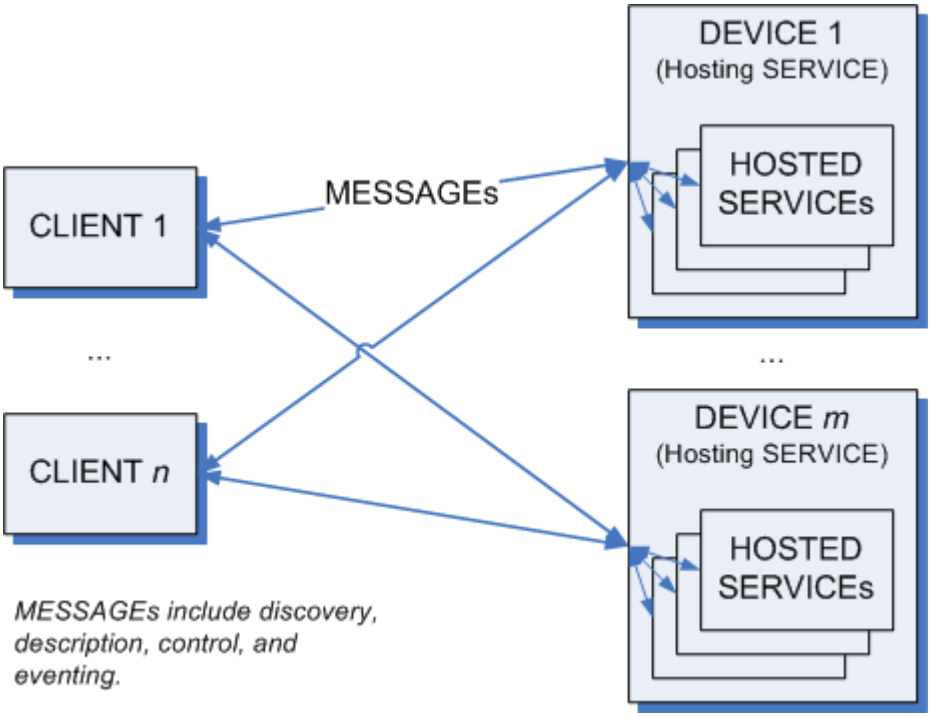
1.1 Requirements

This profile intends to meet the following requirements:

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

2. Terminology and Notation

2.1 Terminology



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

SOAP ENVELOPE
: An XML Infoset that consists of a document information item [XML Infoset] with exactly one member in its **[children]** property, which MUST be the SOAP Envelope [SOAP 1.2] element information item.

MIME SOAP ENVELOPE
: A SOAP ENVELOPE serialized using MIME Multipart Serialization [MTOM].

TEXT SOAP ENVELOPE
: A SOAP ENVELOPE serialized as application/soap+xml.

CLIENT
: A network endpoint that sends MESSAGEs to and/or receives MESSAGEs from a SERVICE.

SERVICE
: A network endpoint that receives and/or sends MESSAGEs to provide a service.

DEVICE
: A distinguished type of SERVICE that hosts other SERVICEs and sends and/or receives one or more specific types of MESSAGEs.

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

SERVICE is visible (not encapsulated) and is addressed separately from its host. Each HOSTED SERVICE has exactly one host. (The relationship is not transitive.)

SENDER
: A CLIENT or SERVICE that sends a MESSAGE.

RECEIVER
: A CLIENT or SERVICE that receives a MESSAGE.

2.2 XML Namespaces

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

```
http://schemas.xmlsoap.org/ws/2005/05/devprof
```

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

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/2005/05/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]

2.3 Notational Conventions

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

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

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

- "?" (0 or 1)
- "*" (0 or more)
- "+" (1 or more)

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

This specification uses the **[action]** and Fault properties [WS-Addressing] to define faults.

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

Rnnn: Normative statement text goes here.

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

2.4 Compliance

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

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

3. Messaging

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

- [SOAP 1.2, Part 1]
- [SOAP 1.2 Part 2, Section 7]
- [SOAP-over-UDP]
- [HTTP/1.1]
- [WS-Addressing]
- [MTOM]

It is assumed that a DEVICE has obtained valid IPv4 and/or IPv6 addresses that do not conflict with other addresses on the network. Mechanisms for obtaining IP addresses are out of the scope of this profile. For more information, see [DHCP] and [IPv6 Autoconfig].

3.1 URI

R0025: A SERVICE MAY fail to process any URI with more than MAX_URI_SIZE octets.
R0027: A SERVICE SHOULD NOT generate a URI with more than MAX_URI_SIZE octets.

The constant MAX_URI_SIZE is defined in Appendix I – Constants.

3.2 UDP

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

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

3.3 HTTP

R0001: A SERVICE MUST support transfer-coding = "chunked".
R0012: A SERVICE MUST at least support the SOAP HTTP Binding.
R0013: A SERVICE MUST at least implement the Responding SOAP Node of the SOAP Request-Response Message Exchange Pattern (http://www.w3.org/2003/05/soap/mep/request-response/).
R0014: A SERVICE MAY choose not to implement the Responding SOAP Node of the SOAP Response Message Exchange Pattern (http://www.w3.org/2003/05/soap/mep/soap-response/).
R0015: A SERVICE MAY choose not to support the SOAP Web Method Feature.

R0014 and R0015 relax requirements in [SOAP 1.2, Part 2, Section 7].

R0030: A SERVICE MUST at least implement the Responding SOAP Node of an HTTP one-way Message Exchange Pattern where the SOAP ENVELOPE is carried in the HTTP Request and the HTTP Response has a Status Code of 202 Accepted and an empty Entity Body (no SOAP ENVELOPE).
R0017: A SERVICE MUST at least support Request Message SOAP ENVELOPEs and one-way SOAP ENVELOPEs that are delivered using HTTP POST.

3.4 SOAP Envelope

R0034: A SERVICE MUST at least receive and send SOAP 1.2 [SOAP 1.2] SOAP ENVELOPEs.
R0003: A SERVICE MAY reject a TEXT SOAP ENVELOPE with more than MAX_ENVELOPE_SIZE octets.
R0026: A SERVICE SHOULD NOT send a TEXT SOAP ENVELOPE with more than MAX_ENVELOPE_SIZE octets.

Large SOAP ENVELOPEs are expected to be serialized using attachments.

3.5 WS-Addressing

*R0004: A DEVICE SHOULD use a uuid scheme URI as the **[address]** property of its Endpoint Reference.*
*R0005: A DEVICE MUST use a stable, globally unique identifier that is constant across network interfaces and IPv4/v6 addresses as the **[address]** property of its Endpoint Reference.*
*R0006: A DEVICE MUST persist the **[address]** property of its Endpoint Reference across re-initialization and changes in the metadata of the DEVICE and any SERVICEs it hosts.*

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

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

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

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

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

*R0031: A SERVICE MAY reject an HTTP Request Message SOAP ENVELOPE if the **[address]** of the **[reply endpoint]** is not "http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous".*
*R0041: If an HTTP Request Message SOAP ENVELOPE generates a SOAP Fault, a SERVICE MAY discard the SOAP Fault if the **[address]** of the **[fault endpoint]** of the HTTP Request Message is not "http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous".*

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

*R0019: A SERVICE MUST include a Message Information Header representing a **[relationship]** property of type wsa:Reply in each Response Message SOAP ENVELOPE the service generates.*
*R0040: A SERVICE MUST include a Message Information Header representing a **[relationship]** property of type wsa:Reply in each SOAP Fault SOAP ENVELOPE the service generates.*

3.6 Attachments

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

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

R0036: A SERVICE MAY reject a MIME SOAP ENVELOPE if the Content-Transfer-Encoding header field mechanism of any MIME part is not "binary".
R0037: A SERVICE MUST NOT send a MIME SOAP ENVELOPE unless the Content-Transfer-Encoding header field mechanism of every MIME part is "binary".

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

R0038: A SERVICE MAY reject a MIME SOAP ENVELOPE if the root part is not the first body part in the Multipart/Related entity.
R0039: A SERVICE MUST NOT send a MIME SOAP ENVELOPE unless root part is the first body part in the Multipart/Related entity.

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

4. Discovery

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

- [WS-Discovery]

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

R1013: A DEVICE MUST be a compliant Target Service.
R1001: A HOSTED SERVICE SHOULD NOT be a Target Service.

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

R1019: A DEVICE MUST at least support the "http://schemas.xmlsoap.org/ws/2005/04/discovery/rfc2396" and "http://schemas.xmlsoap.org/ws/2005/04/discovery/strcmp0" Scope matching rules.
R1020: If a DEVICE includes Types in a Hello, Probe Match, or Resolve Match SOAP ENVELOPE, it MUST include the wsx:MetadataExchange Type.

Including the wsx:MetadataExchange Type indicates a DEVICE supports Get Metadata [WS-MetadataExchange] which is the interoperable means for retrieving metadata about the DEVICE and any HOSTED SERVICEs.

R1009: A DEVICE MUST at least support receiving Probe and Resolve SOAP ENVELOPEs and sending Hello and Bye SOAP ENVELOPEs over multicast UDP.
R1016: A DEVICE MUST at least support sending Probe Match and Resolve Match SOAP ENVELOPEs over unicast UDP.

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

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

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

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

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

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

5. Description

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

- [XML Schema Part 1, Part 2]
- [WSDL 1.1]
- [BP 1.1, Section 4]
- [WSDL Binding for SOAP 1.2]
- [WS-MetadataExchange]
- [WS-Policy]
- [WS-PolicyAttachment]

In highly-constrained circumstances, a CLIENT will know all it needs to know about a DEVICE and its HOSTED SERVICEs to correctly send and receive application-specific MESSAGEs. However, in development scenarios, or when a CLIENT wishes to inspect a DEVICE and take advantage of extended or nonstandard capabilities, a CLIENT will need to retrieve the description (a.k.a. metadata) for a DEVICE and/or its HOSTED SERVICEs.

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

Through WSDL, description also conveys the MESSAGEs a SERVICE is capable of receiving and sending. Through WS-Policy, description conveys the capabilities and

requirements of a SERVICE, particularly the transports over which it may be reached and its security capabilities.

5.1 Characteristics

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

```
<wsdp:ThisModel ...>
  <wsdp:Manufacturer xml:lang="..."? >xs:string</wsdp:Manufacturer>+
  <wsdp:ManufacturerUrl>xs:anyURI</wsdp:ManufacturerUrl>?
  <wsdp:ModelName xml:lang="..."? >xs:string</wsdp:ModelName>+
  <wsdp:ModelNumber>xs:string</wsdp:ModelNumber>?
  <wsdp:ModelUrl>xs:anyURI</wsdp:ModelUrl>?
  <wsdp:PresentationUrl>xs:anyURI</wsdp:PresentationUrl>?
  ...
</wsdp:ThisModel>
```

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

wsdp:ThisModel/ wsdp:Manufacturer
: Name of the manufacturer of the DEVICE. It MUST have fewer than MAX_FIELD_SIZE Unicode characters, SHOULD be localized, and SHOULD be repeated for each supported locale.

wsdp:ThisModel/ wsdp:ManufacturerUrl
: URL to a Web site for the manufacturer of the DEVICE. It MUST have fewer than MAX_URI_SIZE octets.

wsdp:ThisModel/ wsdp:ModelName
: User-friendly name for this model of device chosen by the manufacturer. It MUST have fewer than MAX_FIELD_SIZE Unicode characters, SHOULD be localized, and SHOULD be repeated for each supported locale.

wsdp:ThisModel/ wsdp:ModelNumber
: Model number for this model of DEVICE. It MUST have fewer than MAX_FIELD_SIZE Unicode characters.

wsdp:ThisModel/ wsdp:ModelUrl
: URL to a Web site for this model of DEVICE. It MUST have fewer than MAX_URI_SIZE octets.

wsdp:ThisModel/ wsdp:PresentationUrl
: URL to an HTML page for this DEVICE. It MAY be relative to a base URL and MUST have fewer than MAX_URI_SIZE octets.

CORRECT:

```
<wsdp:ThisModel
    xmlns:wsdp="http://schemas.xmlsoap.org/ws/2005/05/devprof" >
  <wsdp:Manufacturer>ACME Manufacturing</wsdp:Manufacturer>
  <wsdp:ModelName xml:lang="en-GB" >ColourBeam 9</wsdp:ModelName>
  <wsdp:ModelName xml:lang="en-US" >ColorBeam 9</wsdp:ModelName>
</wsdp:ThisModel>

```

A Dialect [WS-MetadataExchange] equal to "http://schemas.xmlsoap.org/ws/2005/05/devprof/ThisModel" indicates an instance of the ThisModel metadata format.

No Identifier [WS-MetadataExchange] is defined for instances of the ThisModel metadata format.

R2038: A DEVICE MUST have one Metadata Section with Dialect equal to "http://schemas.xmlsoap.org/ws/2005/05/devprof/ThisModel" for its ThisModel metadata.

R2012: If no Dialect is specified in a Get Metadata SOAP ENVELOPE, in the corresponding Get Metadata Response SOAP ENVELOPE, a DEVICE MUST include the Metadata Section with Dialect equal to "http://schemas.xmlsoap.org/ws/2005/05/devprof/ThisModel".

Get Metadata [WS-MetadataExchange] is the interoperable means for a CLIENT to retrieve the ThisModel metadata for a DEVICE. A DEVICE may also provide other means for a CLIENT to retrieve its ThisModel metadata.

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

Caching for the ThisModel metadata is controlled by the wsd:MetadataVersion construct [WS-Discovery].

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

```
<wsdp:ThisDevice ...>
  <wsdp:FriendlyName xml:lang="..."? >xs:string</wsdp:FriendlyName>+
  <wsdp:FirmwareVersion>xs:string</wsdp:FirmwareVersion>?
  <wsdp:SerialNumber>xs:string</wsdp:SerialNumber>?
  ...
</wsdp:ThisDevice>
```

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

wsdp:ThisDevice/ wsdp:FriendlyName
: User-friendly name for this DEVICE. It MUST have fewer than MAX_FIELD_SIZE Unicode characters, SHOULD be localized, and SHOULD be repeated for each supported locale.

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

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

CORRECT:

```
<wsdp:ThisDevice
    xmlns:wsdp="http://schemas.xmlsoap.org/ws/2005/05/devprof" >
 <wsdp:FriendlyName xml:lang="en-GB" >
   ACME ColourBeam Printer
 </wsdp:FriendlyName>
 <wsdp:FriendlyName xml:lang="en-US" >
   ACME ColorBeam Printer
 </wsdp:FriendlyName>
</wsdp:ThisDevice>
```

A Dialect [WS-MetadataExchange] equal to "http://schemas.xmlsoap.org/ws/2005/05/devprof/ThisDevice" indicates an instance of the ThisDevice metadata format.

No Identifier [WS-MetadataExchange] is defined for instances of the ThisDevice metadata format.

R2039: A DEVICE MUST have a Metadata Section with Dialect equal to "http://schemas.xmlsoap.org/ws/2005/05/devprof/ThisDevice" for its ThisDevice metadata.
R2014: If no Dialect is specified in a Get Metadata SOAP ENVELOPE, in the corresponding Get Metadata Response SOAP ENVELOPE, a DEVICE MUST include the Metadata Section with Dialect equal to "http://schemas.xmlsoap.org/ws/2005/05/devprof/ThisDevice".

CORRECT:

```
<soap:Envelope
    xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
    xmlns:wsdp="http://schemas.xmlsoap.org/ws/2005/05/devprof"
    xmlns:wsx="http://schemas.xmlsoap.org/ws/2004/09/mex"
    xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing" >
 <soap:Header>
  <wsa:Action>
    http://schemas.xmlsoap.org/ws/2004/09/mex/GetMetadata/Response
  </wsa:Action>
  <wsa:RelatesTo>
    uuid:82204a83-52f6-475c-9708-174fa27659ec
  </wsa:RelatesTo>
  <wsa:To>
    http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous
  </wsa:To>
 </soap:Header>
 <soap:Body>
  <wsx:Metadata>

   <wsx:MetadataSection
Dialect="http://schemas.xmlsoap.org/ws/2005/05/devprof/ThisModel"
   >
    <wsdp:ThisModel>
     <wsdp:Manufacturer>ACME Manufacturing</wsdp:Manufacturer>
     <wsdp:ModelName xml:lang="en-GB" >
       ColourBeam 9
     </wsdp:ModelName>
     <wsdp:ModelName xml:lang="en-US" >
       ColorBeam 9
     </wsdp:ModelName>
    </wsdp:ThisModel>
   </wsx:MetadataSection>

   <wsx:MetadataSection
Dialect="http://schemas.xmlsoap.org/ws/2005/05/devprof/ThisDevice"
   >
    <wsdp:ThisDevice>
     <wsdp:FriendlyName xml:lang="en-GB" >
       ACME ColourBeam Printer
     </wsdp:FriendlyName>
```

```
     <wsdp:FriendlyName xml:lang="en-US" >
       ACME ColorBeam Printer
     </wsdp:FriendlyName>
    </wsdp:ThisDevice>
   </wsx:MetadataSection>

   <!-- Other Metadata Sections omitted for brevity. -->

  </wsx:Metadata>
 </soap:Body>
</soap:Envelope>
```

Get Metadata [WS-MetadataExchange] is the interoperable means for a CLIENT to retrieve the ThisDevice metadata for a DEVICE. A DEVICE may also provide other means for a CLIENT to retrieve its ThisDevice metadata.

R2002: If a DEVICE changes any of its ThisDevice metadata, it MUST increment the Metadata Version exposed in Hello, Probe Match, and Resolve Match SOAP ENVELOPEs as wsd:MetadataVersion.

Caching for the ThisDevice metadata is controlled by the wsd:MetadataVersion construct [WS-Discovery].

5.2 Hosting

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

```
<wsdp:Relationship Type="xs:anyURI" ... >
 (<wsdp:Host>
    <wsa:EndpointReference>endpoint-reference</wsa:EndpointReference>
    <wsdp:Types>list of xs:QName</wsdp:Types>?
    <wsdp:ServiceId>xs:anyURI</wsdp:ServiceId>?
    ...
  </wsdp:Host>)?
 (<wsdp:Hosted>
    <wsa:EndpointReference>endpoint-reference</wsa:EndpointReference>
    <wsdp:Types>list of xs:QName</wsdp:Types>?
    <wsdp:ServiceId>xs:anyURI</wsdp:ServiceId>?
    ...
  </wsdp:Hosted>)*
  ...
</wsdp:Relationship>
```

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

wsdp:Relationship
: Relationship between two or more SERVICEs.

wsdp:Relationship/@Type
: The type of the relationship. This value should be compared directly, as a case-sensitive string, with no attempt to make a relative URI into an absolute URI, to unescape, or to otherwise canonicalize it.

wsdp:Relationship/@Type =
"http://schemas.xmlsoap.org/ws/2005/05/devprof/host"
: Hosting relationship between a HOSTED SERVICE its host. This relationship type defines the following additional content:

wsdp:Relationship/wsdp:Host

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

wsdp:Relationship/wsdp:Host/wsdp:Types

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

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

wsdp:Relationship/wsdp:Host/wsdp:ServiceId

Identifier for the host which MUST be persisted across re-initialization (see also R0005 and R0006) and MAY be shared across multiple Host elements if a host has more than one Endpoint Reference. This value should be compared directly, as a case-sensitive string, with no attempt to make a relative URI into an absolute URI, to unescape, or to otherwise canonicalize it.

wsdp:Relationship/wsdp:Hosted

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

wsdp:Relationship/wsdp:Hosted/wsdp:Types

Unordered set of Types implemented by the HOSTED SERVICE. (See [WS-Discovery].) If omitted, no implied value.

wsdp:Relationship/wsdp:Hosted/wsdp:ServiceId

Identifier for the HOSTED SERVICE which MUST be persisted across re-initialization and MAY be shared across multiple Hosted elements if a HOSTED SERVICE has more than one Endpoint Reference. This identifier allows a CLIENT to recognize which Endpoint References refer to the same HOSTED SERVICE. This value should be compared directly, as a case-sensitive string, with no attempt to make a relative URI into an absolute URI, to unescape, or to otherwise canonicalize it.

CORRECT:

```
<wsdp:Relationship
    Type="http://schemas.xmlsoap.org/ws/2005/05/devprof/host"
    xmlns:img="http://printer.example.org/imaging"
    xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing"
    xmlns:wsdp="http://schemas.xmlsoap.org/ws/2005/05/devprof" >
  <wsdp:Hosted>
    <wsa:EndpointReference>
      <wsa:Address>http://172.30.184.244/print</wsa:Address>
    </wsa:EndpointReference>
    <wsdp:Types>
      img:PrintBasicPortType img:PrintAdvancedPortType
    </wsdp:Types>
```

```
    <wsdp:ServiceId>
      http://printer.example.org/imaging/PrintService
    </wsdp:ServiceId>
  </wsdp:Hosted>
</wsdp:Relationship>
```

A Dialect [WS-MetadataExchange] equal to "http://schemas.xmlsoap.org/ws/2005/05/devprof/Relationship" indicates an instance of the Relationship metadata format.

No Identifier [WS-MetadataExchange] is defined for instances of the Relationship metadata format.

R2040: If a SERVICE has any HOSTED SERVICEs, it MUST have at least one Metadata Section with Dialect equal to "http://schemas.xmlsoap.org/ws/2005/05/devprof/Relationship" for its Relationship metadata.

R2029: If no Dialect is specified in a Get Metadata SOAP ENVELOPE, in the corresponding Get Metadata Response SOAP ENVELOPE, a SERVICE MUST include the Metadata Section(s) with Dialect equal to "http://schemas.xmlsoap.org/ws/2005/05/devprof/Relationship".

Get Metadata [WS-MetadataExchange] is the interoperable means for a CLIENT to retrieve the relationship metadata for a SERVICE. A SERVICE may provide other means for a CLIENT to retrieve its relationship metadata.

CORRECT:

```
<soap:Envelope
    xmlns:gen="http://example.org/general"
    xmlns:img="http://printer.example.org/imaging"
    xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
    xmlns:wsdp="http://schemas.xmlsoap.org/ws/2005/05/devprof"
    xmlns:wsx="http://schemas.xmlsoap.org/ws/2004/09/mex"
    xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing" >
  <soap:Header>
    <wsa:Action>
      http://schemas.xmlsoap.org/ws/2004/09/mex/GetMetadata/Response
    </wsa:Action>
    <wsa:RelatesTo>
      uuid:82204a83-52f6-475c-9708-174fa27659ec
    </wsa:RelatesTo>
    <wsa:To>
      http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous
    </wsa:To>
  </soap:Header>
  <soap:Body>
    <wsx:Metadata>
      <wsx:MetadataSection
Dialect="http://schemas.xmlsoap.org/ws/2005/05/devprof/Relationship"
      >
        <wsdp:Relationship
          Type="http://schemas.xmlsoap.org/ws/2005/05/devprof/host" >
          <wsdp:Hosted>
            <wsa:EndpointReference>
              <wsa:Address>http://172.30.184.244/print</wsa:Address>
            </wsa:EndpointReference>
```

```
            <wsdp:Types>
              img:PrintBasicPortType img:PrintAdvancedPortType
            </wsdp:Types>
            <wsdp:ServiceId>
              http://printer.example.org/imaging/PrintService
            </wsdp:ServiceId>
          </wsdp:Hosted>
          <wsdp:Hosted>
            <wsa:EndpointReference>
              <wsa:Address>http://[fdaa:23]/print1</wsa:Address>
            </wsa:EndpointReference>
            <wsdp:Types>
              img:PrintBasicPortType img:PrintAdvancedPortType
            </wsdp:Types>
            <wsdp:ServiceId>
              http://printer.example.org/imaging/PrintService
            </wsdp:ServiceId>
          </wsdp:Hosted>

          <wsdp:Hosted>
            <wsa:EndpointReference>
              <wsa:Address>http://172.30.184.244/scan</wsa:Address>
            </wsa:EndpointReference>
            <wsdp:Types>img:ScanBasicPortType</wsdp:Types>
            <wsdp:ServiceId>
              http://printer.example.org/imaging/ScanService
            </wsdp:ServiceId>
          </wsdp:Hosted>
          <wsdp:Hosted>
            <wsa:EndpointReference>
              <wsa:Address>http://[fdaa:24]/scan</wsa:Address>
            </wsa:EndpointReference>
            <wsdp:Types>img:ScanBasicPortType</wsdp:Types>
            <wsdp:ServiceId>
              http://printer.example.org/imaging/ScanService
            </wsdp:ServiceId>
          </wsdp:Hosted>
        </wsdp:Relationship>
      </wsx:MetadataSection>

      <!-- Other Metadata Sections omitted for brevity. -->

    </wsx:Metadata>
  </soap:Body>
</soap:Envelope>
```

R2030: If a DEVICE changes any of its relationship metadata, it MUST increment the Metadata Version exposed in Hello, Probe Match, and Resolve Match SOAP ENVELOPEs as wsd:MetadataVersion.

Caching for relationship metadata is controlled by the wsd:MetadataVersion construct [WS-Discovery].

R2042: A DEVICE MUST NOT change its relationship metadata based on temporary changes in the network availability of the SERVICEs described by the metadata.

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

5.3 WSDL

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

R2004 relaxes R2303 in [BP 1.1, Section 4].

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

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

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

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

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

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

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

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

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

For clarity, separation of levels of abstraction, and/or reuse of standardized components, WSDL may be authored in a style that separates different elements of a Service Definition into separate documents which may be imported as needed. Each separate document is included in separate WSDL Metadata Sections.

R2016: If no Dialect is specified in a Get Metadata SOAP ENVELOPE, in the corresponding Get Metadata Response SOAP ENVELOPE, a SERVICE MUST include the Metadata Section(s) with Dialect equal to "http://schemas.xmlsoap.org/wsdl/".

Get Metadata [WS-MetadataExchange] is the interoperable means for a CLIENT to retrieve the WSDL for a HOSTED SERVICE. A HOSTED SERVICE may provide other means for a CLIENT to retrieve its WSDL.

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

CORRECT:

```
<soap:Envelope
    xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
    xmlns:wsx="http://schemas.xmlsoap.org/ws/2004/09/mex"
    xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing" >
  <soap:Header>
    <wsa:Action>
    http://schemas.xmlsoap.org/ws/2004/09/mex/GetMetadata/Response
    </wsa:Action>
    <wsa:RelatesTo>
      uuid:82204a83-52f6-475c-9708-174fa27659ec
    </wsa:RelatesTo>
    <wsa:To>
      http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous
    </wsa:To>
  </soap:Header>
  <soap:Body>
    <wsx:Metadata>

      <wsx:MetadataSection
          Dialect="http://schemas.xmlsoap.org/wsdl" >
        <wsx:MetadataReference>
          <wsa:Address>http://172.30.184.244/print</wsa:Address>
          <wsa:ReferenceParameters>
            <x:Acme xmlns:x="urn:acme.com:webservices">
              WSDL
            </x:Acme>
          </wsa:ReferenceParameters>
        </wsx:MetadataReference>
      </wsx:MetadataSection>

      <!-- Other Metadata Sections omitted for brevity. -->

    </wsx:Metadata>
  </soap:Body>
</soap:Envelope>

```

R2021: If a DEVICE changes its WSDL, it MUST increment the Metadata Version exposed in Hello, Probe Match, and Resolve Match SOAP ENVELOPEs as wsd:MetadataVersion.

Caching for DEVICE WSDL is controlled by the wsd:MetadataVersion construct [WS-Discovery]. Since a HOSTED SERVICE should not be a Target Service, any changes to its WSDL should not be controlled by the wsd:MetadataVersion construct.

5.4 WS-Policy

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

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

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

wsdp:Profile

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

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

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

CORRECT:

```
<wsp:Policy
    xmlns:wsdp="http://schemas.xmlsoap.org/ws/2005/05/devprof"
    xmlns:wsp="http://schemas.xmlsoap.org/ws/2004/09/policy" >
  <wsdp:Profile />
</wsp:Policy>
```

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

To indicate that a SERVICE sends or receives MIME SOAP ENVELOPEs, this profile defines the following WS-Policy assertion:

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

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

wsdp:OptimizedMimeSerialization

A SOAP ENVELOPE MUST be serialized as a MIME SOAP ENVELOPE [MTOM]. This assertion has Endpoint Policy Subject: a policy expression containing this assertion MAY be attached to a wsdl:port, SHOULD be attached to a wsdl:binding, but MUST NOT be attached to a wsdl:portType; the latter is prohibited because the assertion specifies a concrete behavior whereas the wsdl:portType is an abstract construct.

R2011: If a SERVICE supports attachments, the SERVICE MUST include the wsdp:OptimizedMimeSerialization assertion in its policy.

CORRECT:

```
<wsp:Policy
    xmlns:wsdp="http://schemas.xmlsoap.org/ws/2005/05/devprof"
    xmlns:wsp="http://schemas.xmlsoap.org/ws/2004/09/policy" >
  <wsdp:Profile />
  <wsdp:OptimizedMimeSerialization wsp:Optional="true" />
</wsp:Policy>
```

To indicate how a SERVICE supports eventing, this profile defines the following WS-Policy assertions:

```
<wsdp:PushDelivery wsp:Optional="true"? .../>
<wsdp:DurationExpiration wsp:Optional="true"? .../>
<wsdp:ActionFilter wsp:Optional="true"? .../>
```

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

wsdp:PushDelivery

A Subscribe SOAP ENVELOPE MUST use the "http://schemas.xmlsoap.org/ws/2004/08/eventing/DeliveryModes/Push" Delivery Mode [WS-Eventing].

wsdp:DurationExpiration

If a Subscribe or Renew SOAP ENVELOPE includes an Expiration, it MUST be of type xs:duration.

wsdp:ActionFilter

If a Subscribe SOAP ENVELOPE includes a Filter, it MUST use the "http://schemas.xmlsoap.org/ws/2005/05/devprof/Action" Filter Dialect. (See Section 6.1 Subscription.)

These assertions have Endpoint Policy Subject: a policy expression containing these assertion MAY be attached to a wsdl:port, SHOULD be attached to a wsdl:binding, but MUST NOT be attached to a wsdl:portType.

R2032: If a SERVICE exposes Notifications, it MUST include the wsdp:PushDelivery assertion in its policy.
R2033: If a SERVICE exposes Notifications, it MUST include the wsdp:DurationExpiration assertion in its policy.
R2034: If a SERVICE exposes Notifications, it MUST include the wsdp:ActionFilter assertion in its policy.

Including these assertions reflects requirements R3009, 3016, 3017, and R3008. (See Section 6. Eventing.)

CORRECT:

```
<wsp:Policy
    xmlns:wsdp="http://schemas.xmlsoap.org/ws/2005/05/devprof"
    xmlns:wsp="http://schemas.xmlsoap.org/ws/2004/09/policy" >
  <wsdp:Profile />
  <wsdp:OptimizedMimeSerialization wsp:Optional="true" />
  <wsdp:PushDelivery />
  <wsdp:DurationExpiration />
  <wsdp:ActionFilter />
</wsp:Policy>
```

R2041: If a SERVICE uses wsp:PolicyReference/@URI to attach a policy identified by an absolute URI, the SERVICE MUST have a Metadata Section with Dialect equal to "http://schemas.xmlsoap.org/ws/2004/09/policy" and Identifier equal to that URI.
R2025: If a SERVICE uses wsp:PolicyReference/@URI to attach a policy identified by an absolute URI, then, if no Dialect is specified in a Get Metadata SOAP ENVELOPE, in the corresponding Get Metadata Response SOAP ENVELOPE, the SERVICE MUST include the Metadata Section with Dialect equal to "http://schemas.xmlsoap.org/ws/2004/09/policy" and Identifier equal to that URI.
R2035: If a SERVICE uses wsp:PolicyReference/@URI to attach a policy identified by a relative URI, the SERVICE MUST embed that policy as a child of wsdl:definitions, and the policy MUST have a @wsu:Id containing that URI.

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

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

Get Metadata [WS-MetadataExchange] is the interoperable means for a CLIENT to retrieve attached policy.

CORRECT:

```
<soap:Envelope
  xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
  xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
  xmlns:wsdp="http://schemas.xmlsoap.org/ws/2005/05/devprof"
  xmlns:wsoap="http://schemas.xmlsoap.org/wsdl/soap12/"
  xmlns:wsp="http://schemas.xmlsoap.org/ws/2004/09/policy"
  xmlns:wsu
="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-
utility-1.0.xsd"
  xmlns:wsx="http://schemas.xmlsoap.org/ws/2004/09/mex"
  xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing" >
 <soap:Header>
  <wsa:Action>
    http://schemas.xmlsoap.org/ws/2004/09/mex/GetMetadata/Response
  </wsa:Action>
  <wsa:RelatesTo>
    uuid:82204a83-52f6-475c-9708-174fa27659ec
  </wsa:RelatesTo>
  <wsa:To>
    http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous
  </wsa:To>
 </soap:Header>
 <soap:Body>
  <wsx:Metadata>
   <wsx:MetadataSection
     Dialect="http://schemas.xmlsoap.org/wsdl/" >
    <wsdl:definitions
      targetNamespace="http://acme.example.com/colorbeam"
      xmlns:image="http://printer.example.org/imaging" >
     <wsp:Policy wsu:Id="Attachments" >
      <wsdp:Profile />
      <wsdp:OptimizedMimeSerialization wsp:Optional="true" />
      <wsdp:PushDelivery />
      <wsdp:DurationExpiration />
      <wsdp:ActionFilter />
     </wsp:Policy>

     <!-- Other WSDL components omitted for brevity. -->

     <wsdl:service name="MultiFunction" >
      <wsdl:port name="Print" binding="image:PrintBinding" >
       <wsoap:address location="http://172.30.184.244/print" />
       <wsp:PolicyReference URI="#Attachments"
           wsdl:required="true" />
      </wsdl:port>
      <wsdl:port name="Scan" binding="image:ScanBinding" >
       <wsoap:address location="http://172.30.184.244/scan" />
       <wsp:PolicyReference URI="#Attachments"
```

```
            wsdl:required="true" />
        </wsdl:port>
       </wsdl:service>
     </wsdl:definitions>
    </wsx:MetadataSection>

    <!-- Other Metadata Sections omitted for brevity. -->

    </wsx:Metadata>
  </soap:Body>
</soap:Envelope>
```

R2022: If a DEVICE changes its Policy, it MUST increment the Metadata Version exposed in Hello, Probe Match, and Resolve Match SOAP ENVELOPEs as wsd:MetadataVersion.

Caching for DEVICE Policy is controlled by the wsd:MetadataVersion construct [WS-Discovery]. Since a HOSTED SERVICE should not be a Target Service, any changes to its Policy should not be controlled by the wsd:MetadataVersion construct.

6. Eventing

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

- [WS-Eventing]

6.1 Subscription

R3009: A SERVICE MUST at least support Push Delivery Mode indicated by "http://schemas.xmlsoap.org/ws/2004/08/eventing/DeliveryModes/Push".
R3010: A SERVICE MUST NOT generate a wse:DeliveryModeRequestedUnavailable SOAP Fault in response to a Subscribe SOAP ENVELOPE with a Delivery Mode of "http://schemas.xmlsoap.org/ws/2004/08/eventing/DeliveryModes/Push".

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

*R3017: If a SERVICE does not understand the **[address]** of the Notify To of a Subscribe SOAP ENVELOPE, the SERVICE MUST generate a wsa:DestinationUnreachable SOAP Fault.*
*R3018: If a SERVICE does not understand the **[address]** of the End To of a Subscribe SOAP ENVELOPE, the SERVICE MUST generate a wsa:DestinationUnreachable SOAP Fault.*
R3019: If a SERVICE cannot deliver a Notification SOAP ENVELOPE to an Event Sink, the SERVICE MAY terminate the corresponding Subscription and SHOULD send a Subscription End SOAP ENVELOPE with a Status of "http://schemas.xmlsoap.org/ws/2004/08/eventing/DeliveryFailure".

6.1.1 Filtering

To enable subscribing to one or more Notifications exposed by a SERVICE, this profile defines a Filter Dialect designated "http://schemas.xmlsoap.org/ws/2005/05/devprof/Action".

- A Filter in this Dialect contains a white space-delimited list of URIs that indicate the **[action]** property of desired Notifications.
- The content of a Filter in this Dialect is defined as xs:list/@itemType="*xs:anyURI*" [XML Schema Part 2].
- A Filter in this Dialect evaluates to true for an Output Message of a Notification or Solicit-Response operation if and only if a URI in the Filter matches the **[action]** property of the Message using the "http://schemas.xmlsoap.org/ws/2005/04/discovery/rfc2396" matching rule [WS-Discovery].

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

R3008: A SERVICE MUST at least support Filtering by the Dialect "http://schemas.xmlsoap.org/ws/2005/05/devprof/Action".

CORRECT:

```
<soap:Envelope
  xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
  xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing"
  xmlns:wse="http://schemas.xmlsoap.org/ws/2004/08/eventing" >
 <soap:Header>
  <wsa:Action>
    http://schemas.xmlsoap.org/ws/2004/08/eventing/Subscribe
  </wsa:Action>
  <wsa:MessageID>
    uuid:314bea3b-03af-47a1-8284-f495497f1e33
  </wsa:MessageID>
  <wsa:ReplyTo>
   <wsa:Address>
    http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous
   </wsa:Address>
  </wsa:ReplyTo>
  <wsa:To>http://172.30.184.244/print</wsa:To>
 </soap:Header>
 <soap:Body>
  <wse:Subscribe>
   <wse:Delivery>
    <wse:NotifyTo>
     <wsa:Address>
       uuid:3726983d-02de-4d41-8207-d028ae92ce3d
     </wsa:Address>
    </wse:NotifyTo>
   </wse:Delivery>
   <wse:Expires>PT10M</wse:Expires>
   <wse:Filter
Dialect="http://schemas.xmlsoap.org/ws/2005/05/devprof/Action"
   >
http://printer.example.org/imaging/PrintBasicPortType/JobEndState
http://printer.example.org/imaging/PrintBasicPortType/PrinterState
   </wse:Filter>
```

```
  </wse:Subscribe>
 </soap:Body>
</soap:Envelope>
```

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

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

R3012: A SERVICE MUST NOT generate a wse:FilteringRequestedUnavailable SOAP Fault in response to a Subscribe SOAP ENVELOPE with a Filter Dialect of "http://schemas.xmlsoap.org/ws/2005/05/devprof/Action".

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

[action]	http://schemas.xmlsoap.org/ws/2005/05/devprof/Fault
[Code]	soap:Sender
[Subcode]	wsdp:FilterActionNotSupported
[Reason]	E.g., "no notifications match the supplied filter"
[Detail]	(None defined.)

*R3020: If none of the Notifications exposed by a SERVICE match the **[action]** values in a Subscribe SOAP ENVELOPE Filter whose Dialect is "http://schemas.xmlsoap.org/ws/2005/05/devprof/Action", the SERVICE MUST generate a wsdp:FilterActionNotSupported SOAP Fault.*

6.2 Subscription Duration and Renewal

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

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

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

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

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

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

Event Sources are not required to support retrieving subscription status.

7. Security

This section defines a RECOMMENDED baseline for interoperable security between a DEVICE and a CLIENT. A DEVICE (or CLIENT) is free to support other security mechanisms in addition to, or in place of, this mechanism as specified by WSDL [WSDL 1.1], policies [WS-Policy], or other mechanisms. In the absence of an explicit indication stating that a different security mechanism is to be used, the default security mechanism defined here is assumed to apply.

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

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

- [AES/TLS]
- [HTTP Authentication]
- [SHA1]
- [TLS]
- [UUID]
- [X.509.v3]

7.1 Secure Communication

7.1.1 Integrity

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

R4000: A SERVICE MUST not send a SOAP ENVELOPE without protecting the integrity of any Message Information Header blocks matching the following XPath expressions: (a) /soap:Envelope/soap:Header/wsa:Action, (b) /soap:Envelope/soap:Header/wsa:MessageID, (c) /soap:Envelope/soap:Header/wsa:To, (d) /soap:Envelope/soap:Header/wsa:ReplyTo, (e) /soap:Envelope/soap:Header/wsa:RelatesTo.
R4063: A SERVICE MAY reject a SOAP ENVELOPE that has unprotected Message Information Header blocks.
R4001: A SERVICE MUST not send a SOAP ENVELOPE without protecting the integrity of the SOAP ENVELOPE Body in conjunction with any Message Information Block(s) from R4000.
R4064: A SERVICE MAY reject a SOAP ENVELOPE that does not protect the integrity of the SOAP ENVELOPE Body.

In this profile, the integrity of discovery SOAP ENVELOPEs is protected using message-level signatures, while the integrity of other MESSAGEs is protected using a

Secure Channel. Other profiles may use alternate mechanisms to protect the integrity of MESSAGEs.

7.1.2 Confidentiality

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

R4002: A SERVICE MUST NOT send a SOAP ENVELOPE without encrypting the SOAP ENVELOPE Body.
R4067: A SERVICE MAY reject a SOAP ENVELOPE that does not encrypt the SOAP ENVELOPE Body.
R4003: A SENDER MUST provide key transfer information to authorized RECEIVERs.

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

7.1.3 Authentication

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

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

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

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

R4005: On multicast MESSAGEs, a CLIENT MUST use an authentication credential that is suitable for all DEVICEs that could legitimately process the multicast MESSAGE.

7.1.4 Trust

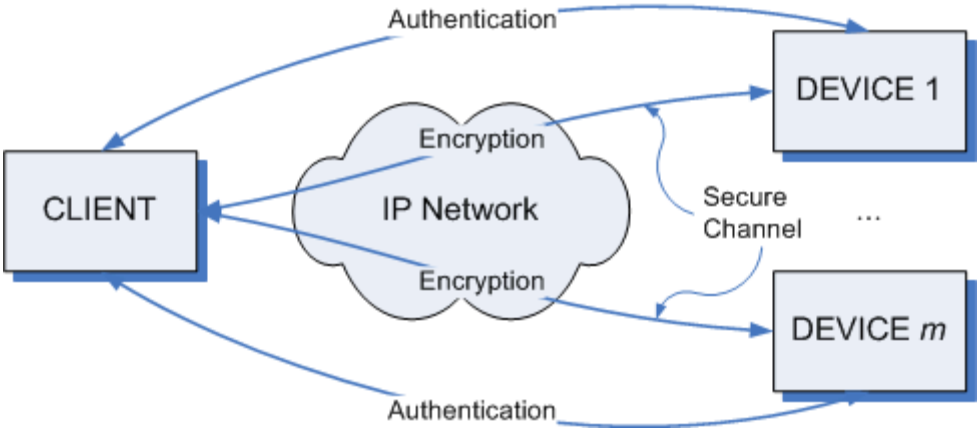
There are different trust models associated with DEVICE security. The following requirements profile the kinds of trust that may be used with DEVICE security in this profile.

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

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

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

7.1.5 Network Model



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

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

It is assumed that MESSAGEs received from/via other administrative domains cannot be trusted.

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

7.1.6 Security Association

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

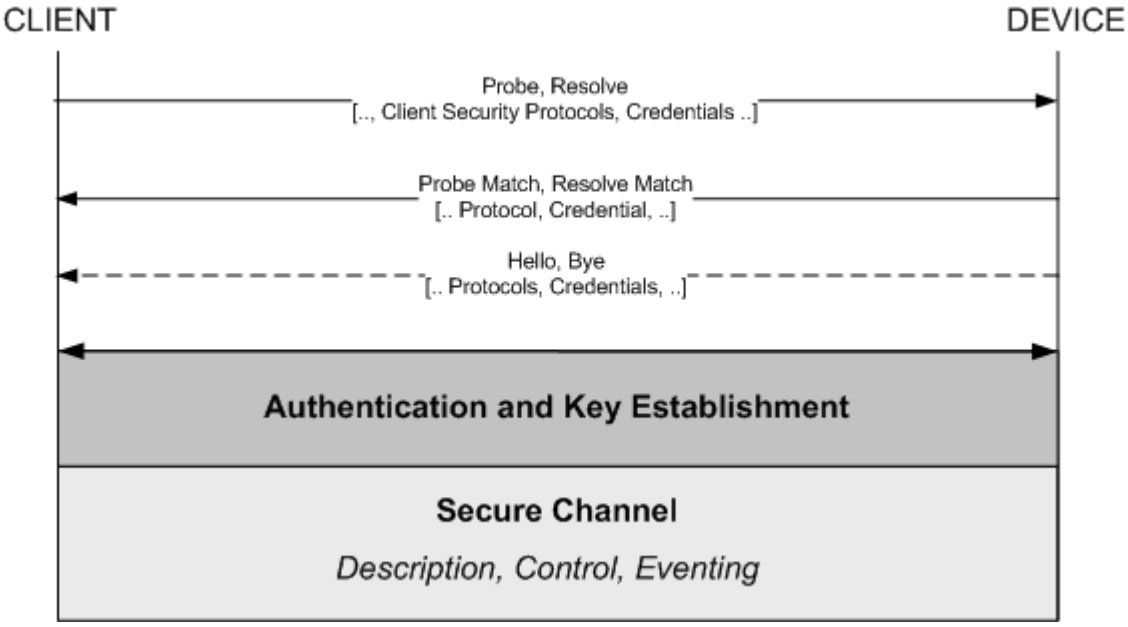
The supported protocols for authentication and key establishment are advertised and negotiated during discovery.

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

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

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

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



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

7.1.7 DEVICE Behavior

R4014: A DEVICE MAY require authentication of a CLIENT.
R4015: To verify the authenticity of multicast messages sent by the DEVICE during discovery, i.e., Hello and Bye SOAP ENVELOPEs, multicast MESSAGEs SHOULD be signed.
R4016: Unicast MESSAGEs sent by a DEVICE in response to multicast MESSAGEs, i.e., Probe Match and Resolve Match SOAP ENVELOPEs, SHOULD be signed.
R4017: A CLIENT MAY ignore MESSAGEs received during discovery that have no signature or a nonverifiable signature.
R4018: A DEVICE SHOULD cache authentication information for a CLIENT as valid as long as the DEVICE is connected to the CLIENT.

7.1.8 Security Protocols and Credentials

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

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

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

R4027: A CLIENT MUST use the Security Protocol and Credential indicated by the DEVICE in the Probe Match (or Resolve Match) SOAP ENVELOPE for authentication and key establishment.
R4028: CLIENTs and DEVICEs SHOULD support the following Security Protocols and Credentials for authentication and key establishment: TLS with client certificates and server certificates, respectively.
R4069: CLIENTs and DEVICEs MUST support HTTP Basic Authentication.

7.1.9 Security for Discovery

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

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

Probe

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

R4030: A Probe SOAP ENVELOPE SHOULD contain the Security Protocols and Credentials in /soap:Envelope/ soap:Header/ wsa:ReplyTo/ wsp:Policy.
R4031: In the absence of any policy assertion for security, no security SHALL be required.
R4032: A Device MUST NOT send a Probe Match SOAP ENVELOPE if any of the following are true: (a) the DEVICE is outside the local subnet of the CLIENT, and the Probe SOAP ENVELOPE was sent as multicast, or (b) the DEVICE does not support the indicated CLIENT Security Protocols and Credentials.
R4065: A CLIENT MUST discard a Probe Match SOAP ENVELOPE if it is received MATCH_TIMEOUT seconds or more later than the last corresponding Probe SOAP ENVELOPE was sent.

Hello

R4034: A DEVICE SHOULD sign a Hello SOAP ENVELOPE.

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

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

Resolve

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

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

Bye

R4037: A DEVICE SHOULD sign a Bye SOAP ENVELOPE.

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

7.1.10 Authentication

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

Transport Layer Security (TLS)

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

DEVICE Authentication with TLS

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

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

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

Certificates

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

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

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

1313 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 761661ccba7546a09d5f03a8ea3b1160 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

1314 The Subject field (listed above) contains the UUID in string representation format.

1315 Certificate management is out of the scope of this profile.

1316 TLS Authentication with Client Certificate

R4071: If the CLIENT and the DEVICE exchanged certificates during the TLS handshake, and the DEVICE as well as the CLIENT were able to verify the certificates, the CLIENT and DEVICE are mutually authenticated, and no further steps SHALL be required.
R4046: A DEVICE MAY require an additional authentication step after the TLS handshake, if the DEVICE was not able to verify the certificate, or if the CLIENT did not provide a certificate during the TLS handshake.
R4047: A DEVICE MAY require HTTP Authentication.
R4048: If the HTTP authentication is successful, and the CLIENT presents a certificate to the DEVICE, the DEVICE SHOULD cache the certificate in its local certificate store of trusted certificates for future authentication of the CLIENT.

This avoids the need for HTTP authentication for subsequent associations.

HTTP Authentication

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

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

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

R4050: If a DEVICE requires HTTP authentication, the DEVICE SHALL challenge the CLIENT using the HTTP 401 response code.
R4051: A CLIENT MUST authenticate using one of the options listed in the HTTP-Authenticate header.
R4052: HTTP Authentication MUST use the following parameters for username and password of the HTTP Request: UserName, PIN / Password.

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

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

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

R4054: The RECOMMENDED size of a PIN / Password is at least 8 characters using at least a 32 character alphabet.
R4055: The PIN / Password that is unique to the DEVICE SHALL be conveyed to the CLIENT out-of-band. The methods of conveying the PIN out-of-band are out of the scope of this profile.
R4056: To reduce the attack surface, the DEVICE and CLIENT MAY limit the number of failed authentication attempts as well as the time interval successive attempts are made for one TLS session.

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

7.1.11 Secure Channel

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

R4057: All secure communication for Description, Control, and Eventing between the CLIENT and DEVICE MUST use the Secure Channel. The protocols for encryption as well as the keys used for encryption are negotiated during the authentication phase.
R4072: A DEVICE MUST support receiving and responding to a Probe SOAP ENVELOPE over HTTP using the Secure Channel.
R4073: A DEVICE MAY ignore a Probe SOAP ENVELOPE sent over HTTP that does not use the Secure Channel.

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

7.1.12 TLS Ciphersuites

R4059: It is the responsibility of the sender to convert the embedded URL to use HTTPS as different transport security mechanisms can be negotiated.
R4060: A DEVICE MUST support the following TLS Ciphersuite: TLS_RSA_WITH_RC4_128_SHA.
R4061: It is recommended that a DEVICE also support the following TLS Ciphersuite: TLS_RSA_WITH_AES_128_CBC_SHA.
R4062: Additional Ciphersuites MAY be supported. They are negotiated during the TLS handshake.

7.2 Security Policy Assertions

This profile defines the following assertions to indicate the Security Protocols.

```
<wsdp:Tls wsp:Optional="true"? ... />
<wsdp:X509Cert wsp:Optional="true"? ... />
```

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

wsdp:Tls
: This assertion indicates the SERVICE (or CLIENT) requires TLS.

wsdp:X509Cert
: This assertion indicates the SERVICE (or CLIENT) requires X.509 certificates for authentication.

These assertions have Endpoint Policy Subject [WS-PolicyAttachment]: a policy expression containing one of these assertions MAY be attached to a wsdl:port, SHOULD be attached to a wsdl:binding, but MUST NOT be attached to a wsdl:portType; the latter is prohibited because these assertions specify concrete behaviors whereas the wsdl:portType is an abstract construct.

8. Acknowledgements

This profile has been developed as a result of joint work with many individuals and teams, including: Don Box (Microsoft), Mike Fenelon (Microsoft), Bertus Greeff

(Microsoft), Rob Hain (Microsoft), Rich Hasha (Microsoft), Gopal Kakivaya (Microsoft), Chris Kurt (Microsoft), David Lindsey (Lexmark), Jonathan Marsh (Microsoft), Sam Rhodus (Lexmark), Adam Sapek (Microsoft), Stacy Simpson (Lexmark), Lifen Tian (Ricoh), David Turner (Microsoft), Mike Vernal (Microsoft), Yaotian Wang (Ricoh), Kenny Wolf (Microsoft).

9. References

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

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

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

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

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

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

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

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

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

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

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

[UUID]
P. Leach, et al, "A UUID URN Namespace," December 2004. (See http://www.ietf.org/internet-drafts/draft-mealling-uuid-urn-05.txt.)

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

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

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

[WSDL Binding for SOAP 1.2]
K. Ballinger, et al, "WSDL Binding for SOAP 1.2," April 2002. (See http://groups.yahoo.com/group/soapbuilders/files/soap12WSDL.htm.)

[WS-Eventing]
L. Cabrera, et al, "Web Services Eventing (WS-Eventing)," August 2004. (See http://msdn.microsoft.com/ws/2004/08/ws-eventing/.)

[WS-MetadataExchange]
K. Ballinger, et al, "Web Services Metadata Exchange (WSMetadataExchange)," September 2004. (See http://msdn.microsoft.com/ws/2004/09/ws-metadataexchange/.)

[WS-Policy]
S. Bajaj, et al, "Web Services Policy Framework (WS-Policy)," September 2004. (See http://schemas.xmlsoap.org/ws/2004/09/policy.)

[WS-PolicyAttachment]
S. Bajaj, et al, "Web Services Policy Attachment (WS-PolicyAttachment)," September 2004. (See http://schemas.xmlsoap.org/ws/2004/09/policy.)

[WS-Security 2004]
A. Nadalin, et al, "Web Services Security: SOAP Message Security 1.0 (WS-Security 2004)," March 2004. (See http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-soap-message-security-1.0.pdf.)

[X.509.v3]
"ITU-T X.509.v3 Information technology - Open Systems Interconnection - The Directory: Public-key and attribute certificate frameworks (ISO/IEC/ITU 9594-8)."

[XML Schema, Part 1]
H. Thompson, et al, "XML Schema Part 1: Structures," May 2001. (See http://www.w3.org/TR/2001/REC-xmlschema-1-20010502/.)

[XML Schema, Part 2]
P. Biron, et al, "XML Schema Part 2: Datatypes," May 2001. (See http://www.w3.org/TR/2001/REC-xmlschema-2-20010502/.)

10. Informative References

The following documents are referenced for informational purposes only. They are not part of the scope of the profile:

[IPv6 Autoconfig]
S. Thomson, et al, "IPv6 Stateless Address Autoconfiguration," December 1998. (See http://www.ietf.org/rfc/rfc2462.txt.)

[DHCP]
R. Droms, "Dynamic Host Configuration Protocol," March 1997. (See http://www.ietf.org/rfc/rfc2131.txt.)

[RFC 2119]
S. Bradner, "Key words for use in RFCs to Indicate Requirement Levels," RFC 2119, March 1997. (See http://www.ietf.org/rfc/rfc2119.txt.)

[XML Infoset]
J. Cowan, et al, "XML Information Set (Second Edition)," February 2004. (See http://www.w3.org/TR/2004/REC-xml-infoset-20040204/.)

Appendix I – Constants

The following constants are used throughout this profile. The values listed below 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]

Appendix II – XML Schema

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

http://schemas.xmlsoap.org/ws/2005/05/devprof/devicesprofile.xsd

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

```
<xs:schema
    targetNamespace="http://schemas.xmlsoap.org/ws/2005/05/devprof"
    xmlns:tns="http://schemas.xmlsoap.org/ws/2005/05/devprof"
    xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing"
    xmlns:xs="http://www.w3.org/2001/XMLSchema"
    elementFormDefault="qualified"
    blockDefault="#all" >

  <xs:import
    namespace="http://schemas.xmlsoap.org/ws/2004/08/addressing"
    schemaLocation
    ="http://schemas.xmlsoap.org/ws/2004/08/addressing/addressing.xsd"
```

```
    />

  <xs:element name="ThisModel" type="tns:ThisModelType" />
  <xs:complexType name="ThisModelType" >
    <xs:sequence>
      <xs:element name="Manufacturer" type="tns:LocalizedStringType"
          maxOccurs="unbounded" />
      <xs:element name="ManufacturerUrl" type="xs:anyURI"
          minOccurs="0" />
      <xs:element name="ModelName" type="tns:LocalizedStringType"
          maxOccurs="unbounded" />
      <xs:element name="ModelNumber" type="xs:string" minOccurs="0" />
      <xs:element name="ModelUrl" type="xs:anyURI" minOccurs="0" />
      <xs:element name="PresentationUrl" type="xs:anyURI"
          minOccurs="0" />
      <xs:any namespace="##other" processContents="lax"
              minOccurs="0" maxOccurs="unbounded" />
    </xs:sequence>
    <xs:anyAttribute namespace="##other" processContents="lax" />
  </xs:complexType>

  <xs:element name="ThisDevice" type="tns:ThisDeviceType" />
  <xs:complexType name="ThisDeviceType" >
    <xs:sequence>
      <xs:element name="FriendlyName" type="tns:LocalizedStringType"
          maxOccurs="unbounded" />
      <xs:element name="FirmwareVersion" type="xs:string"
          minOccurs="0" />
      <xs:element name="SerialNumber" type="xs:string" minOccurs="0" />
      <xs:any namespace="##other" processContents="lax"
              minOccurs="0" maxOccurs="unbounded" />
    </xs:sequence>
    <xs:anyAttribute namespace="##other" processContents="lax" />
  </xs:complexType>

  <xs:complexType name="LocalizedStringType" >
    <xs:simpleContent>
      <xs:extension base="xs:string" >
        <xs:anyAttribute namespace="##other" processContents="lax" />
      </xs:extension>
    </xs:simpleContent>
  </xs:complexType>

  <xs:element name="Relationship" >
    <xs:complexType>
      <xs:sequence>
        <xs:any namespace="##any" processContents="lax"
                minOccurs="0" maxOccurs="unbounded" />
      </xs:sequence>
      <xs:attribute name="Type" type="xs:anyURI" use="required" />
      <xs:anyAttribute namespace="##other" processContents="lax" />
    </xs:complexType>
  </xs:element>

  <xs:element name="Host" type="tns:HostServiceType" />
  <xs:element name="Hosted" type="tns:HostServiceType" />
  <xs:complexType name="HostServiceType" >
```

```
    <xs:sequence>
      <xs:element ref="wsa:EndpointReference" />
      <xs:element ref="tns:Types" minOccurs="0" />
      <xs:element ref="tns:ServiceId" minOccurs="0" />
      <xs:any namespace="##other" processContents="lax"
              minOccurs="0" maxOccurs="unbounded" />
    </xs:sequence>
    <xs:anyAttribute namespace="##other" processContents="lax" />
  </xs:complexType>

  <xs:element name="ServiceId" type="xs:anyURI" />
  <xs:element name="Types" type="tns:QNameListType" />
  <xs:simpleType name="QNameListType" >
    <xs:list itemType="xs:QName" />
  </xs:simpleType>

  <xs:element name="Profile" type="tns:AssertionType" />
  <xs:element name="OptimizedMimeSerialization"
      type="tns:AssertionType" />
  <xs:element name="PushDelivery" type="tns:AssertionType" />
  <xs:element name="DurationExpiration" type="tns:AssertionType" />
  <xs:element name="ActionFilter" type="tns:AssertionType" />
  <xs:element name="Tls" type="tns:AssertionType" />
  <xs:element name="X509Cert" type="tns:AssertionType" />

  <xs:complexType name="AssertionType" >
    <xs:complexContent>
      <xs:restriction base="xs:anyType">
        <xs:anyAttribute namespace="##other" processContents="lax" />
      </xs:restriction>
    </xs:complexContent>
  </xs:complexType>

</xs:schema>
```