6LoWAPP
Internet-Draft
Intended status: Informational
Expires: June 21, 2010

G. Moritz
University of Rostock

December 18, 2009

DPWS for 6LoWPAN
draft-moritz-6lowapp-dpws-enhancements-00

Abstract

This draft describes adaptions and enhancements for deploying the Devices Profile for Web Service (DPWS) in 6LoWPAN networks.

Status of this Memo

This Internet-Draft is submitted to IETF in full conformance with the provisions of BCP 78 and BCP 79. This document may not be modified, and derivative works of it may not be created, and it may not be published except as an Internet-Draft.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

The list of current Internet-Drafts can be accessed at http://www.ietf.org/ietf/1id-abstracts.txt.

The list of Internet-Draft Shadow Directories can be accessed at http://www.ietf.org/shadow.html.

This Internet-Draft will expire on June 21, 2010.

Copyright Notice

Table of Contents

1. Introduction

In August 2008 a technical committee (TC) at OASIS was formed for the Web Services Discovery and Web Services Devices Profile(WS-DD) [WS-DD]. WS-DD standardizes a lightweight subset of the Web services protocol suite that makes it easy to find, share, and control devices on a network.

The work of this TC is based on the former DPWS, WS-Discovery, and SOAP-over-UDP specifications. DPWS makes use of existing Web services protocols, but also adds several extensions to enable Web services based communication on embedded devices also. Thereby, DPWS includes features like (1) discovery of devices and metadata exchange with services even in dynamic changing environments (2) eventing about state changes by WS-Eventing (3) security and integrity for discovery, metadata exchange and service usages. Because DPWS bases on existing Web services standards, it is fully capable of being integrated in the Web services framework.

This draft describes several adaptions and enhancements to expand DPWS deployments to 6LoWPAN networks, but is far away from a comprehensive specification. It only presents a basis for further discussions. Main scope is the definition of a profile, to describe: message compression and bidirectional message reduction, while staying fully compliant with existing WS-DD specifications. The deployment of this profile is fully transparent for existing DPWS implementations and describes extension to be considered by 6LoWPAN networks only.

Readers of this draft should have a basic knowledge about the specifications DPWS [DPWS], WS-Discovery [WS-Discovery], SOAP-over-UDP [SOAP-over-UDP] and related standards like SOAP, HTTP, XML and XML Schema.

1.1. Requirements Language

The key words "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 [RFC2119].

1.2. Terminology

DPWS
: In the remainder of this draft, DPWS is used as general term for the WS-DD specifications DPWS [DPWS], WS-Discovery [WS-Discovery], and SOAP-over-UDP [SOAP-over-UDP].

DPWS specification
: Points to the DPWS [DPWS] specification directly.

Device
: A device is an endpoint implementing DPWS device functionalities as specified by WS-DD [WS-DD].

Client
: A client is an endpoint implementing DPWS client functionalities as specified by WS-DD [WS-DD].

Hello
: The Hello message of a device as defined in WS-Discovery [WS-Discovery].

Bye
: The Bye message of a device as defined in WS-Discovery [WS-Discovery].

Probe
: The Probe message of a client as defined in WS-Discovery [WS-Discovery].

Probe Match
: The Probe Match message of a device as defined in WS-Discovery [WS-Discovery].

Resolve
: The Resolve message of a client as defined in WS-Discovery

[WS-Discovery].

Resolve Match
: The Resolve Match message of a device as defined in WS-Discovery [WS-Discovery].

WSDL
: Acronym for the document describing the services in Web Services Description Language [WSDL] format.

Edge Router
: Edge Routers are the routers that connect LoWPANs to an IPv6 infrastructure via backhaul or backbone links when such an infrastructure is available. (Defined in 6LoWPAN Neighbor Discovery [I-D.ietf-6lowpan-nd]

2. Discovery optimizations

DPWS describes two different modes for discovery of devices: ad-hoc mode and managed mode. In managed mode, a registry called Discovery Proxy is applied to suppress multicast messages. This section describes adaptions for both of these modes.

2.1. General adaptions

A DEVICE MUST include the transport specific addresses in its Hello and Probe Match messages.

In accordance to DPWS, embedding of a transport specific address in Hello and Probe messages is not mandatory. This behavior is counterproductive for WSN, with the constraints for energy consumption and limited bandwidth. Thus, the optional fields for the transport specific addresses are now mandatory to avoid Resolve messages.

A DEVICE SHOULD include all device types in Hello and Probe Match messages.

In the current version of DPWS it is not mandatory to include the type field in the Hello and Probe Match messages. A client MAY infer to services provided by the device with the help of the device type. Inclusion of the device type can avoid further discovery and metadata exchange messages.

A SERVICE SHOULD NOT provide the WSDL file for CLIENTS at run time.

Providing the WSDL during the discovery phase is optional in DPWS. WSDL are used in general at development time only for code generation. These WSDL files have a size of several kB in most analyzed scenarios. The expensive and memory consuming storage of these WSDL files on the device and on the client node is not applicable for WSN. Furthermore, the exchange itself consumes a high bandwidth.

2.2. Discovery addressing

- All SOAP-over-UDP messages inside the 6LoWPAN network MUST use the port 61616 as target port. (Exact port to be defined)

- Devices inside the 6LoWPAN network MUST listen to the IPv6 multicast address: FF02::C0. (Exact address to be defined)

- Clients inside the 6LoWPAN network MUST listen to the IPv6 multicast address: FF02::C1. (Exact address to be defined)

In RFC4944, a UDP port compression is described which works most efficient when using ports in a specific range. Thus, the used ports should be changed to fit in this range. The ports have to be mapped on the edge router of the 6LoWPAN network for incoming and outgoing SOAP-over-UDP messages.

DPWS defines one IPv4 and one IPv6 multicast addresses to be used for discovery message exchange. But DPWS differentiates between device and client roles. Usage of one and the same address for addresses exclusively to be processed by clients or devices implies overhead in sending and receiving these messages to all DPWS nodes independent of their role. Inside 6LoWPAN networks, different addresses have to be used. The mapping into compliant addresses is done by the edge router of the 6LoWPAN network.

For a transparent integration, in ad-hoc mode, edge routers of 6LoWPAN networks SHOULD forward incoming and outgoing link-local scope multicast discovery messages.

2.3. Discovery proxy

In managed mode, a device and service registry is applied. It is possible, to use more than one discovery proxy in a network. In managed mode, one discovery proxy SHOULD be deployed at the edge router to hide expensive external multicast messages from the 6LoWPAN network and omit multicast flooding.

2.4. Heartbeat message

Wireless Sensor Networks are unreliable due to low power radio communication and limited battery capacities. Clients and discovery proxies might use a heartbeat message to ask for a device and its status. This heartbeat message can use a pull exchange pattern or a push mechanism similar to eventing functionalities. DPWS defines Directed Probe and includes WS-Eventing, which might fulfill the requirements of the heartbeat message or if a new message has to be defined. The device MUST answer to this heartbeat signal with a unicast Hello including the WS-Discovery Metadata Version indicating state changes of the device or the services.

The definition of the heartbeat mechanisms may be out of scope of a specification and might be included in a domain specific profile (e.g. healthcare scenarios). But the mechanism is required by the device registry described in this draft.

2.5. Device registry

For simple services as provided in 6LoWPAN networks (basic sensors), device metadata messages are almost the most bandwidth consuming messages. Metadata of a device (Model, Manufacturer, Version Number, etc.) of a device are stable in most cases over lifetime of a device. A device registry located at the edge router of a 6LoWPAN network MAY store information about device metadata exchanged through this edge

router by sniffing messages. External metadata exchanged requests to devices in the 6LoWPAN network MAY be answered representative by the device registry. This is transparent to the devices and the clients. The device registry is a hidden intermediate in contrast to the discovery proxy. To provide end-to-end reliability and a guaranty about correct data for the response, the device registry SHOULD invoke the device by using the heartbeat message mechanism described in this draft, before answering the client. The heartbeat message and the response are much smaller compared to the device metadata messages.

3. Message compression

Because DPWS bases on SOAP and thus on XML for data representation, XML compression techniques and/or encoding concepts have to be used to reduce message sizes.

3.1. HTTP compression

DPWS for 6LoWPAN requires HTTP header compression. While CHOPAN [I-D.frank-6lowapp-chopan] describes a general and generic HTTP compression, this draft focuses on a more DPWS specific compression scheme as described here.

DPWS uses SOAP-over-HTTP binding for unicast messaging. All messages are using the POST method of HTTP in version 1.1. The transport specific addresses (target host) can be elided and derived from transport layer. In accordance to HTTP 1.1, all connection marked not explicit as close are keep-alive connections. But keep-alive connections are not applicable for WSN. The SOAPAction field is mandatory when using the SOAP-over-HTTP binding, but can be empty. Because DPWS includes usage of WS-Addressing, the SOAPAction field is redundant. The content-type of SOAP-over-HTTP is always application/soap+xml; charset=utf-8. To sum up, only few fields are left, which are analyzed by devices and clients and which provide useful information. A specific compression scheme is required to omit unnecessary HTTP header fields and allow a compression (optimized binary format) of the remaining required fields. The fields which have to be left are:

Content-length: This is required to allow resource constrained sensor

nodes to know about length of data to be analyzed and thus e.g. required memory allocations.

HTTP Status Codes: The status code may be analyzed in error and fault cases. Status code 200/OK can be implied if this field is missing, to use it only in error/fault cases.

3.2. SOAP compression

Different XML specific and XML non-specific compression schemes are already known. The following table presents an overview about existing schemes and compressors, including the EXI format of the W3C. The table shows resulting sizes of different compressors applied to DPWS messages. The values present the sizes of the SOAP envelopes (excluding HTTP headers) after compression and in the last line pure XML messages for comparison. The messages were recorded in a realistic scenario, implemented by using compliant DPWS toolkits. An overall number of 18 different message types are included in the evaluations and the table shows the averages over all these message types. Most of the compressors suffer from the simple XML structures. Sensor nodes will not provide complex services and thus simple message have to be assumed. These measurements might provide a basis for further discussions on message size optimizations.

For the measurements of EXI schema-informed format, separated results are presented: optimized and default. The default format used XML schema files as defined in DPWS without optimizations. This includes an inconsistency of different namespaces and versions used by DPWS and among Web services specifications (especially WS-Addressing and WS-Eventing). For the optimized format, some improvements are added. Most values of the XML tags and attributes in DPWS are well-known URIs. If these values are included in the XML schema files and with corrected dependencies, the average message size is reduced significantly as presented in the table.

Compressor	Average in Byte	Average in %	Minimum in Byte	Maximum in Byte
EXI schema-informed (optimized)	153,72	19,97	66,00	349,00
EXI schema-informed (default)	206,61	26,49	122,00	415,00
EXI schema-less	315,67	40,31	192,00	633,00
gzip (C=9)	419,83	54,54	271,00	749,00
XMLPPM	427,44	55,61	274,00	755,00
gzip (C=1)	437,83	56,96	297,00	799,00
Xmill (C=9)	457,89	59,46	300,00	824,00
Xmill (C=1)	463,56	60,14	303,00	852,00
bzip2 (C=1)	472,94	61,41	304,00	852,00
bzip2 (C=9)	474,78	61,82	315,00	852,00
Fast Infoset	561,89	69,70	315,00	1301,00
XML	814,89	100,00	418,00	2089,00

Table 1: DPWS SOAP envelope compression comparison

3.3. Compression integration

This section describes a general point, which might be discussed more in general in 6LoWAPP.

3.3.1. Payload compression

Many protocols (like DPWS) already provide concepts for discovery of devices (ad-hoc networking), data dissemination, eventing, etc. 6LoWPAN protocols allow a seamless connectivity of IP networks with wireless sensor networks, without the need for application layer gateways. These gateways must be aware of application layer data and need an understanding of semantics of payload. The communication with existing networking devices or other existing implementations must be transparent for these external hosts. Application layer data compression and encoding should only affect the 6LoWPAN network and communication inside the network, like 6LoWPAN does with IPv6 headers. But payload on application layer is not self-contained in one packet like IP, TCP and UDP headers. Defining new extension to be implemented by the external nodes is not a proper solution and violates the core concept of 6LoWPAN protocols.

New possibilities for application layer data encoding must be found, to allow efficient data encoding for traffic inside the 6LoWPAN

network without affecting external communication. A new Encoding Router (ER) role might be defined. This ER is located at the edge router and not only translates compressed network and transport layer protocols, but also standardized application layer encoding and compression (e.g. EXI format in compliant XML/SOAP envelopes). This requires no understanding of semantics of the payload, but allows a seamless connectivity. The deployment of an ER might violate the layered model, because the ER must receive external message as representative to the 6LoWPAN nodes, encodes the messages and forwards them (outgoing traffic vice versa). But protocols might require correct transport layer addresses for origin and target hosts. Thus, adaptions to transport layer header fields (TCP and UDP) are required at runtime to hide the transparent intermediate ER.

3.3.2. TCP vs. UDP

TCP makes use of mechanisms like sliding window and flow control, to optimize throughput. These mechanisms are questionable in wireless sensor networks. Hence, the above described Encoding Router (ER) might also allow an throughput optimized external communication all the way to the ER and more optimized mechanisms in the 6LoWPAN networks. To reduce TCP overhead, also UDP might be used inside 6LoWPAN networks instead. But most application layer protocols base on TCP because of the required end-to-end reliability. The usage of the lightweight UDP for internal communication instead of TCP would require additional mechanisms to assure end-to-end reliability between endpoints. Also definition of an extension for UDP to provide this functionality is possible, but reinventing TCP must be omitted.

4. IANA Considerations

The new defined discovery addresses have to be registered at IANA.

5. Security Considerations

No security issues have been identified in this draft.

6. References

6.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

6.2. Informative References

[DPWS] Driscoll and Mensch, "OASIS Devices Profile for Web Services (DPWS) Version 1.1", 2009, <http://docs.oasis-open.org/ws-dd/ns/dpws/2009/01>.

[EXI-format] Scheider and Kamiya, "W3C Efficient XML Interchange (EXI) Format 1.0 Candidate Recommendation", December 2009, <http://www.w3.org/TR/2009/CR-exi-20091208/>.

[I-D.frank-6lowapp-chopan] Frank, B., "Chopan - Compressed HTTP Over PANs", draft-frank-6lowapp-chopan-00 (work in progress), September 2009.

[I-D.ietf-6lowpan-nd] Shelby, Z., Thubert, P., Hui, J., Chakrabarti, S., Bormann, C., and E. Nordmark, "6LoWPAN Neighbor Discovery", draft-ietf-6lowpan-nd-07 (work in progress), October 2009.

[SOAP-over-UDP] Jeyaraman, "OASIS SOAP-over-UDP Version 1.1", 2009, <http://docs.oasis-open.org/ws-dd/soapoverudp/1.1/os/wsdd-soapoverudp-1.1-spec-os.html>.

[WS-DD] OASIS Open, "OASIS Web Services Discovery and Web Services Devices Profile (WS-DD) TC", 2009, <www.oasis-open.org/committees/ws-dd/>.

[WS-Discovery]

Modi and Kemp, "OASIS Web Services Dynamic Discovery (WS-Discovery) Version 1.1", 2009, <http://docs.oasis-open.org/ws-dd/ns/discovery/2009/01>.

[WSDL] Christensen, Curbera, Meredith, and Weerawarana, "W3C Web Services Description Language (WSDL) 1.1", March 2001, <http://www.w3.org/TR/2009/CR-exi-20091208/>.

Author's Address

Guido Moritz
University of Rostock
18051 Rostock,
Germany

Phone: +49 381 498 7269
Email: guido.moritz@uni-rostock.de