BPC SML Profile Version 1.0

January 4, 2022

Business Payments Coalition

Table of Contents

1 Version History....3
2 Introduction....3
2.1 Scope....3
2.2 Conformance....3
2.3 Terms and Definitions....3
2.4 Disclaimers and Copyright....3
3 Use of DNS....4
3.1 SML DNS domain name....4
3.2 Constructing the DNS name....4
3.3 The NAPTR record....4
3.4 Time to live (TTL)....6
4 Appendix A: Python DNS name construction example (non-normative)....6
5 Appendix B: Example shell script to return SMP service from SML query (non-normative) 7

1 Version History

Revision Date	Version	Change Description	Editor Name
11/22/2021	0.9	Initial import into template	Britta Holland
1/4/2022	1.0	Incorporated IOC feedback	Britta Holland

2 Introduction

2.1 Scope

This specification is a profile of the Business Document Metadata Service Location (BDXL) Version 1.0 OASIS Standard (OASIS BDXL 1.0) published here: http://docs.oasis-open.org/bdxr/BDX-Location/v1.0/BDX-Location-v1.0.html.

This specification describes how Access Points must query the BPC network Service Metadata Locator (SML) to discover SMP services of Participants. The interoperability and other technical interfaces between network registrars, and between SML Service Providers, and the DNS server storage of the records is specified elsewhere and not in the scope of this specification.

2.2 Conformance

The keywords 'MUST', 'MUST NOT', 'REQUIRED', 'SHALL', 'SHALL NOT', 'SHOULD', 'SHOULD NOT', 'RECOMMENDED', 'MAY', and 'OPTIONAL' in this specification are to be interpreted as described in RFC2119 and RFC 8174 when, and only when, they appear in all capitals, as shown here.

2.3 Terms and Definitions

For the purpose of this specification, all terms shall have the definitions defined in section 2.3 of the *E-invoice Exchange Framework – Approach to Managing a Federated Registry Services Model in a Four-Corner Network* report found here: https://businesspaymentscoalition.org/wp-content/uploads/bpc-e-delivery-network-validation-exercise-2020.pdf

2.4 Disclaimers and Copyright

Views expressed here are not necessarily those of, and should not be attributed to, any particular BPC participant or organization. They are not intended to provide business or legal advice, nor are they intended to promote or advocate a specific action, payment strategy, or product. Readers should consult with their own business and legal advisors.

This specification is the work product of the BPC, and readers are free to republish this specification in whole or in part without further permission, as long as the work is attributed to the BPC, and in no way suggests the BPC sponsors, endorses or recommends any organization or its services or products. Other product names and company names

referenced within this document may be either trademarks or service marks of their respective owners.

<Add MIT licensing statement here.>

3 Use of DNS

3.1 SML DNS domain name

The DNS domain name used for the BPC SML is:

bpcb2b.net

All Participants in the network SHALL be discoverable through querying this domain. Access Point services MUST only use this domain for Participant Discovery.

3.2 Constructing the DNS name

To retrieve the SML record containing information about a Participant's SMP service, an SML client application must first know the complete party identifier and party identifier scheme of the Participant being queried. The format of the party identifier and party identifier scheme MUST follow the BPC Identifier Policy specification.

The DNS name MUST be constructed as follows:

1. The complete party identifier scheme and the complete party identifier MUST be concatenated and delimited by two colons ("::") as follows:

   `{Lowercase identifier scheme}::{Lowercase party identifier},`

2. The SHA256 hash of the *lowercased* concatenated string is computed,
3. The SHA256 digest is Base32 encoded, removing any trailing "=" characters,
4. The final Base32 encoded string is used as the hostname component together with the SML DNS domain name.

For example, a company identified by the GLN number 123456789 would use the ebCore PartyId type scheme identifer `urn:oasis:names:tc:ebcore:partyid-type:iso6523:0060`. The scheme identifier is then concatenated with the party identifier to produce the string:

`urn:oasis:names:tc:ebcore:partyid-type:iso6523:0060::123456789`

The Base32 encoding of the SHA256 hash of the above (lowercased) string is:

`64yibi3w4xki6uspe6lxnmdaxto3eezhpd3uqweb7sybmyvfrnha====`

This string, stripped from its trailing "=" characters and used with the DNS domain name becomes the complete DNS name:

`64yibi3w4xki6uspe6lxnmdaxto3eezhpd3uqweb7sybmyvfrnha.sml.example.com`

See also Appendix A: Python DNS name construction example (non-normative).

3.3 The NAPTR record

As described in OASIS BDXL 1.0, the SML stores the SMP service endpoint information in DNS NAPTR records. To discover the SMP service endpoint of a given Participant, one must query the SML by first constructing a DNS name as described in section 2.2 and then retrieve its NAPTR records from the DNS.

A NAPTR record consists of the following fields:

```
order   preference   flags   service   reg.exp.   replacement
```

Querying a Participant in the SML will return a structure similar to the following example:

```
IN NAPTR 100 10 "U" "oasis-bdxr-smp-2"
    "!^.*$!https://smp.example.com/myservice/!"
```

The above example shows a NAPTR record with these fields and values:

```
Order       100
Preference  10
Flags       "U"
Service     "oasis-bdxr-smp-2"
Reg.exp.    "!^.*$!https://smp.example.com/myservice/!"
Replacement none (the NAPTR record contains an empty string)
```

The trailing "." denotes the end of the NAPTR record.

The fields of the NAPTR record SHALL have the following definition, meaning and use in the BPC network:

Field	Definition and use
Order	No Participant can be registered in the SML with the same service value in more than one NAPTR record. The Order and Preference fields therefore have no practical use. An SML client MAY ignore these fields.
Preference	
Flags	The SML NAPTR records MUST always be of type "U", meaning that the value of the flag field MUST be set to either "U" (uppercase) or "u" (lowercase). The value of the flag field MUST be treated as case insensitive.
Service	This field describes which type of service the record points to. The value of the service field MUST be an identifier defined in a network service profile or specification. Currently only the OASIS SMP 2.0 service is defined. An entity MUST NOT be registered with more than one record having the same value in the service field. The value of the service field MUST be treated as case insensitive.

Reg.exp.	The value of the reg.exp. field is a "!"-delimited string containing a substitution expression consisting of a regular expression and URL:
	`!{regular expression}!{URL}!1`
	The `URL` MUST be an absolute URL to the SMP service of the Participant being queried.
	The `regular expression` MUST be exactly
	`^.*$`
	SMP clients SHOULD ignore the `regular expression` and MAY assume that the `URL` contains the entire fully qualified URL to the SMP service.
	The value of the reg.exp. field MUST be treated as case sensitive.

Replacement	The replacement field of the NAPTR record MUST be empty. SMP clients SHOULD ignore this field.

See also Appendix B: Example shell script to return SMP service from SML query (non-normative).

3.4 Time to live (TTL)

It is RECOMMENDED that SML services set the time to live (TTL) of DNS records to no less than 3,600 seconds (one hour) and to no more than 86,400 seconds (one day).

Access Points MUST NOT cache the DNS record for any longer than the specified TTL.

4 Appendix A: Python DNS name construction example (non-normative)

```
# DNS domain name
DOMAIN = 'sml.example.com'

# import libraries
from hashlib import sha256
from base64 import b32encode

# company identified by the GLN number 123456789:
id = '123456789'
scheme = 'urn:oasis:names:tc:ebcore:partyid-type:iso6523:0060'

# concatenation of scheme and id
# produces the string: urn:oasis:names:tc:ebcore:partyid-
    type:iso6523:0060::123456789
party = scheme + '::' + id

# base32 encode the sha256 hash of the lowercased party string and strip
    trailing '='s
# produces the string: 64YIBI3W4XKI6USPE6LXNMDAXTO3EEZHPD3UQWEB7SYBMYVFRNHA
hostname = b32encode(sha256(party.lower()).digest()).rstrip('=')

# construct the full DNS name
dns_name = hostname + '.' + DOMAIN
```

```
# prints:
    64YIBI3W4XKI6USPE6LXNMDAXTO3EEZHPD3UQWEB7SYBMYVFRNHA.sml.example.com
print dns_name
```

5 Appendix B: Example shell script to return SMP service from SML query (non-normative)

```
#!/bin/bash

# The calculated DNS name of the Participant being queried
DNS_NAME=64yibi3w4xki6uspe6lxnmdaxto3eezhpd3uqweb7sybmyvfrnha.sml.example.com

# Queries the DNS and returns only the line containing the SMP service
# Produces the string:
# 64yibi3w4xki6uspe6lxnmdaxto3eezhpd3uqweb7sybmyvfrnha.sml.example.com. 3600 IN
    NAPTR 100 100 "U" "oasis-bdxr-smp-2"
    "!^.*$!https://smp.example.com/myservice/!" .
NAPTR=`dig -t NAPTR $DNS_NAME | grep 'oasis-bdxr-smp-2'`

# Exits script if no service is found
if [ -z "$NAPTR" ]
then
   echo "No record exists for participant"
   exit
fi

# Return only the reg.exp. field from the NAPTR string
# Produces the string: "!^.*$!https://smp.example.com/myservice/!
REGEX=`echo $NAPTR | grep -o '!^.*!'`

# Assumes that applying the regular expression is not necessary and that the
    string
# part contains the entire URL.
# Splits the string using "!" as delimiter and returns the string part
# Produces the string: https://smp.example.com/myservice/!"
RESULT=`echo $REGEX | cut -d'!' -f3`

    echo $RESULT
```