Stream: Internet Engineering Task Force (IETF)
RFC: 8790
Category: Standards Track
Published: June 2020
ISSN: 2070-1721
Authors: A. Keränen M. Mohajer
Ericsson

RFC 8790
FETCH and PATCH with Sensor Measurement Lists (SenML)

Abstract

The Sensor Measurement Lists (SenML) media type and data model can be used to send collections of resources, such as batches of sensor data or configuration parameters. The Constrained Application Protocol (CoAP) FETCH, PATCH, and iPATCH methods enable accessing and updating parts of a resource or multiple resources with one request. This document defines new media types for the CoAP FETCH, PATCH, and iPATCH methods for resources represented using the SenML data model.

Status of This Memo

This is an Internet Standards Track document.

This document is a product of the Internet Engineering Task Force (IETF). It represents the consensus of the IETF community. It has received public review and has been approved for publication by the Internet Engineering Steering Group (IESG). Further information on Internet Standards is available in Section 2 of RFC 7841.

Information about the current status of this document, any errata, and how to provide feedback on it may be obtained at https://www.rfc-editor.org/info/rfc8790.

Copyright Notice

Table of Contents

1. Introduction

The Sensor Measurement Lists (SenML) media type [RFC8428] and data model can be used to transmit collections of resources, such as batches of sensor data or configuration parameters.

An example of a SenML collection is shown below:

```
[
 {"bn":"2001:db8::2/3311/0/", "n":"5850", "vb":true},
 {"n":"5851", "v":42},
 {"n":"5750", "vs":"Ceiling light"}
]
```

Here, three resources, "3311/0/5850", "3311/0/5851", and "3311/0/5750", of a dimmable light smart object [IPSO] are represented using a single SenML Pack with three SenML Records. All resources share the same base name "2001:db8::2/3311/0/"; hence, full names for the resources are "2001:db8::2/3311/0/5850", etc.

The CoAP [RFC7252] FETCH, PATCH, and iPATCH methods [RFC8132] enable accessing and updating parts of a resource or multiple resources with one request.

This document defines two new media types, one using the JavaScript Object Notation (JSON) [RFC8259] and one using the Concise Binary Object Representation (CBOR) [RFC7049], which can be used with the CoAP FETCH, PATCH, and iPATCH methods for resources represented using the SenML data model (i.e., for both SenML and Sensor Streaming Measurement Lists (SenSML) data). The rest of the document uses the term "(i)PATCH" when referring to both methods as the semantics of the new media types are the same for the CoAP PATCH and iPATCH methods.

2. Terminology

The key words "**MUST**", "**MUST NOT**", "**REQUIRED**", "**SHALL**", "**SHALL NOT**", "**SHOULD**", "**SHOULD NOT**", "**RECOMMENDED**", "**NOT RECOMMENDED**", "**MAY**", and "**OPTIONAL**" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

Readers should also be familiar with the terms and concepts discussed in [RFC8132] and [RFC8428]. The following additional terms are used in this document:

Fetch Record: One set of parameters that is used to match SenML Record(s).

Fetch Pack: One or more Fetch Records in an array structure.

Patch Record: One set of parameters similar to Fetch Record but also containing instructions on how to change existing SenML Pack(s).

Patch Pack: One or more Patch Records in an array structure.

Target Record: A Record in a SenML Pack that matches the selection criteria of a Fetch or Patch Record and hence is a target for a Fetch or Patch operation.

Target Pack: A SenML Pack that is a target for a Fetch or Patch operation.

(i)PATCH: A term that refers to both CoAP "PATCH" and "iPATCH" methods when there is no difference in this specification as to which one is used.

3. Using FETCH and (i)PATCH with SenML

The FETCH/(i)PATCH media types for SenML are modeled as extensions to the SenML media type to enable reuse of existing SenML parsers and generators, in particular on constrained devices. Unless mentioned otherwise, FETCH and PATCH Packs are constructed with the same rules and constraints as SenML Packs.

The key differences from the SenML media type are allowing the use of a "null" value for removing Records with the (i)PATCH method and the lack of value fields in Fetch Records. Also, the Fetch and Patch Records do not have a default time or base version when the fields are omitted.

3.1. SenML FETCH

The FETCH method can be used to select and return a subset of Records, in sequence, of one or more SenML Packs. The SenML Records are selected by giving a set of names that, when resolved, match resolved names in a Target SenML Pack. The names for a Fetch Pack are given using the SenML "name" and/or "base name" fields. The names are resolved by concatenating the base name with the name field as defined in [RFC8428].

A Fetch Pack **MUST** contain at least one Fetch Record. A Fetch Record **MUST** contain a name and/or base name field.

For example, to select the resources "5850" and "5851" from the example in Section 1, the following Fetch Pack can be used:

```
[
 {"bn":"2001:db8::2/3311/0/", "n":"5850"},
 {"n":"5851"}
]
```

The result of a FETCH request with the example above would be:

```
[
 {"bn":"2001:db8::2/3311/0/", "n":"5850", "vb":true},
 {"n":"5851", "v":42},
]
```

The SenML time and unit fields can be used in a Fetch Record to further narrow the selection of matched SenML Records. When no time or unit is given in a Fetch Record, all SenML Records with the given name are matched (i.e., unlike with SenML Records, the lack of time field in a Fetch Record does not imply a time value of zero). When time is given in the Fetch Record, a Target Record is matched only when its resolved time value and name are equal to those of the

Fetch Record. Similarly, when unit is given, a Target Record is matched only when its resolved unit and name are equal to those of the Fetch Record. If both the time and unit are given in the Fetch Record, a Target Record is matched only when both are equal to those of the Fetch Record. Each Target Record **MUST** be included in the response at most once, even if multiple Fetch Records match with the same Target Record.

For example, if the resource "5850" had multiple sensor readings (SenML Records) with different time values, the following Fetch Pack can be used to retrieve the Record with time "1.276020091e+09":

```
[
 {"bn":"2001:db8::2/3311/0/", "n":"5850", "t":1.276020091e+09}
]
```

The resolved form of Records (Section 4.6 of [RFC8428]) is used when comparing the names, times, and units of the Target and Fetch Records to accommodate differences in the use of the base values. In the resolved form, the SenML name in the example above becomes "2001:db8::2/3311/0/5850". Since there is no base time in the Pack, the time in resolved form is equal to the time in the example.

If no SenML Records match, an empty SenML Pack (i.e., array with no elements) is returned as a response.

Fetch Records **MUST NOT** contain other fields than name, base name, time, base time, unit, and base unit. Implementations **MUST** reject and generate an error for a Fetch Pack with other fields. [RFC8132], Section 2.2 provides guidance for FETCH request error handling, e.g., using the 4.22 (Unprocessable Entity) CoAP error response code.

3.2. SenML (i)PATCH

The (i)PATCH method can be used to change the fields of SenML Records, to add new Records, and to remove existing Records. The names, times, and units of the Patch Records are given and matched in the same way as for the Fetch Records, except each Patch Record **MUST** match at most one Target Record. A Patch Record matching more than one Target Record is considered invalid (patching multiple Target Records with one Patch Record would result in multiple copies of the same Record). Patch Packs can also include new values and other SenML fields for the Records. Application of Patch Packs is idempotent; hence, the PATCH and iPATCH methods for SenML Packs are equivalent.

When the name in a Patch Record matches with the name in an existing Record, the resolved time values and units (if any) are compared. If the time values and units either do not exist in both Records or are equal, the Target Record is replaced with the contents of the Patch Record. All Patch Records **MUST** contain at least a SenML Value or Sum field.

If a Patch Record contains a name, or the combination of a time value, unit, and name, that does not exist in any existing Record in the Pack, the given Record, with all the fields it contains, is added to the Pack.

If a Patch Record has a value ("v") field with a null value, it **MUST NOT** be added, but the matched Record (if any) is removed from the Target Pack.

The Patch Records **MUST** be applied in the same sequence as they are in the Patch Pack. If multiple Patch Packs are being processed at the same time, the result **MUST** be equivalent to applying them in one sequence.

Implementations **MUST** reject and generate an error for Patch Packs with invalid Records. If a Patch Pack is rejected, the state of the Target Pack is not changed, i.e., either all or none of the Patch Records are applied. [RFC8132], Section 3.4 provides guidance for error handling with PATCH and iPATCH requests, e.g., using the 4.22 (Unprocessable Entity) and 4.09 (Conflict) CoAP error response codes.

For example, the following document could be given as an (i)PATCH payload to change/set the values of two SenML Records for the example in Section 1:

```
[
 {"bn":"2001:db8::2/3311/0/", "n":"5850", "vb":false},
 {"n":"5851", "v":10}
]
```

If the request is successful, the resulting representation of the example SenML Pack would be as follows:

```
[
 {"bn":"2001:db8::2/3311/0/", "n":"5850", "vb":false},
 {"n":"5851", "v":10},
 {"n":"5750", "vs":"Ceiling light"}
]
```

As another example, the following document could be given as an (i)PATCH payload to remove the two SenML Records:

```
[
 {"bn":"2001:db8::2/3311/0/", "n":"5850", "v":null},
 {"n":"5851", "v":null}
]
```

4. Fragment Identification

Fragment identification for Records of Fetch and Patch Packs uses the same mechanism as SenML JSON/CBOR fragment identification (see Section 9 of [RFC8428]), i.e., the "rec" scheme followed by a comma-separated list of Record positions or range(s) of Records. For example, to select the 3rd and 5th Record of a Fetch or Patch Pack, a fragment identifier "rec=3,5" can be used in the URI of the Fetch or Patch Pack resource.

5. Extensibility

The SenML mandatory-to-understand field extensibility mechanism (see Section 4.4 of [RFC8428]) does not apply to Patch Packs, i.e., unknown fields **MUST NOT** generate an error, but such fields are treated like any other field (e.g., added to Patch target Records where applicable).

This specification allows only a small subset of SenML fields in Fetch Records, but future specifications may enable new fields for Fetch Records and possibly also new fields for selecting targets for Patch Records.

6. Security Considerations

The security and privacy considerations of SenML also apply to the FETCH and (i)PATCH methods. CoAP's security mechanisms are used to provide security for the FETCH and (i)PATCH methods.

In FETCH and (i)PATCH requests, the client can pass arbitrary names to the target resource for manipulation. The resource implementer must take care to only allow access to names that are actually part of (or accessible through) the target resource. In particular, the receiver needs to ensure that any input does not lead to uncontrolled special interpretation by the system.

If the client is not allowed to do a GET or PUT on the full target resource (and thus all the names accessible through it), access control rules must be evaluated for each Record in the Pack.

7. IANA Considerations

This document registers two new media types and CoAP Content-Format IDs for both media types.

7.1. CoAP Content-Format Registration

IANA has assigned CoAP Content-Format IDs for the SenML PATCH and FETCH media types in the "CoAP Content-Formats" subregistry, within the "Constrained RESTful Environments (CoRE) Parameters" registry [RFC7252]. The assigned IDs are shown in Table 1.

Media Type	Encoding	ID
application/senml-etch+json	-	320
application/senml-etch+cbor	-	322

Table 1: CoAP Content-Format IDs

7.2. senml-etch+json Media Type

Type name: application

Subtype name: senml-etch+json

Required parameters: N/A

Optional parameters: N/A

Encoding considerations: binary

Security considerations: See Section 6 of RFC 8790.

Interoperability considerations: N/A

Published specification: RFC 8790

Applications that use this media type: Applications that use the SenML media type for resource representation.

Fragment identifier considerations: Fragment identification for application/senml-etch+json is supported by using fragment identifiers as specified by Section 4 of RFC 8790.

Additional information:

Deprecated alias names for this type: N/A

Magic number(s): N/A

File extension(s): senml-etchj

Windows Clipboard Name: "SenML FETCH/PATCH format"

Macintosh file type code(s): N/A

Macintosh Universal Type Identifier code:
org.ietf.senml-etch-json conforms to public.text

Person & email address to contact for further information:
Ari Keränen <ari.keranen@ericsson.com>

Intended usage: COMMON

Restrictions on usage: N/A

Author: Ari Keränen <ari.keranen@ericsson.com>

Change controller: IESG

7.3. senml-etch+cbor Media Type

Type name: application

Subtype name: senml-etch+cbor

Required parameters: N/A

Optional parameters: N/A

Encoding considerations: binary

Security considerations: See Section 6 of RFC 8790.

Interoperability considerations: N/A

Published specification: RFC 8790

Applications that use this media type: Applications that use the SenML media type for resource representation.

Fragment identifier considerations: Fragment identification for application/senml-etch+cbor is supported by using fragment identifiers as specified by Section 4 of RFC 8790.

Additional information:

Deprecated alias names for this type: N/A

Magic number(s): N/A

File extension(s): senml-etchc

Macintosh file type code(s): N/A

Macintosh Universal Type Identifier code:
org.ietf.senml-etch-cbor conforms to public.data

Person & email address to contact for further information:
Ari Keränen <ari.keranen@ericsson.com>

Intended usage: COMMON

Restrictions on usage: N/A

Author: Ari Keränen <ari.keranen@ericsson.com>

Change controller: IESG

8. References

8.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC7049] Bormann, C. and P. Hoffman, "Concise Binary Object Representation (CBOR)", RFC 7049, DOI 10.17487/RFC7049, October 2013, <https://www.rfc-editor.org/info/rfc7049>.

[RFC7252] Shelby, Z., Hartke, K., and C. Bormann, "The Constrained Application Protocol (CoAP)", RFC 7252, DOI 10.17487/RFC7252, June 2014, <https://www.rfc-editor.org/info/rfc7252>.

[RFC8132] van der Stok, P., Bormann, C., and A. Sehgal, "PATCH and FETCH Methods for the Constrained Application Protocol (CoAP)", RFC 8132, DOI 10.17487/RFC8132, April 2017, <https://www.rfc-editor.org/info/rfc8132>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

[RFC8259] Bray, T., Ed., "The JavaScript Object Notation (JSON) Data Interchange Format", STD 90, RFC 8259, DOI 10.17487/RFC8259, December 2017, <https://www.rfc-editor.org/info/rfc8259>.

[RFC8428] Jennings, C., Shelby, Z., Arkko, J., Keranen, A., and C. Bormann, "Sensor Measurement Lists (SenML)", RFC 8428, DOI 10.17487/RFC8428, August 2018, <https://www.rfc-editor.org/info/rfc8428>.

8.2. Informative References

[IPSO] IPSO, "IPSO Light Control Smart Object", 2019, <http://www.openmobilealliance.org/tech/profiles/lwm2m/3311.xml>.

Acknowledgements

The use of the FETCH and (i)PATCH methods with SenML was first introduced by the OMA SpecWorks Lightweight Machine to Machine (LwM2M) v1.1 specification. This document generalizes the use to any SenML representation. The authors would like to thank Carsten Bormann, Christian Amsüss, Jaime Jiménez, Klaus Hartke, Michael Richardson, and other participants from the IETF CoRE and OMA SpecWorks DMSE working groups who have contributed ideas and reviews.

Authors' Addresses

Ari Keränen
Ericsson
FI-02420 Jorvas
Finland
Email: ari.keranen@ericsson.com

Mojan Mohajer
Email: mojanm@hotmail.com