Workgroup: HTTP
Internet-Draft: draft-beky-httpbis-metadata-00
Published: 8 July 2022
Intended Status: Standards Track
Expires: 9 January 2023
Authors: B. Béky
Google LLC
B. Roy
Google LLC

METADATA frame for HTTP/2 and HTTP/3

Abstract

This document describes a mechanism to send meta information over HTTP/2 ([RFC9113]) and HTTP/3 ([RFC9114]) connections that refers to either the entire connection or a specific stream without changing the semantics of the HTTP messages. This mechanism can be used, for example, to gather information for accounting or logging purposes.

Discussion Venues

This note is to be removed before publishing as an RFC.

Discussion of this document takes place on the HTTP Working Group mailing list (ietf-http-wg@w3.org), which is archived at https://lists.w3.org/Archives/Public/ietf-http-wg/.

Source for this draft and an issue tracker can be found at https://github.com/bencebeky/metadata.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

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."

This Internet-Draft will expire on 9 January 2023.

Copyright Notice

Table of Contents

1. Introduction

HTTP/2 and HTTP/3 connections are capable of transporting multiple HTTP messages, which are composed of field sections and bodies. This document describes a mechanism to convey additional information about HTTP messages or the entire connection, in a way that does not change HTTP semantics, over the same connection. For instance, an endpoint may wish to convey the CPU cost or other loadbalancing information for a particular HTTP message, or perhaps certain statistics for a particular HTTP message or for the connection as a whole. Applications may wish to provide such information without affecting HTTP messages themselves. These are some non-exhaustive examples of use cases that may be well served by the METADATA frame.

A proxy ***MAY*** consume METADATA frames, pass them along unmodified, modify the payloads, or emit new METADATA frames, depending on the specific needs of the application.

2. Conventions and Definitions

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.

3. METADATA frame

Both HTTP/2 and HTTP/3 specifications allow the protocol to be extended, see Section 5.5 of [RFC9113] and Section 9 of [RFC9114].

This document defines a new frame type: METADATA.

The payload of a sequence of METADATA frames is a metadata block, which is an encoded list of key-value pairs. Each key and value is a sequence of bytes with no restriction on the allowed values.

An endpoint ***MAY*** transmit multiple metadata blocks on the same stream.

METADATA frames do not change HTTP semantics.

3.1. METADATA HTTP/2 frame

The type of METADATA HTTP/2 frame is 0x4d.

```
METADATA HTTP/2 Frame {
  Length (24),
  Type (8) = 0x4d,

  Flags (8),

  Reserved(1),
  Stream Identifier (31),

  Encoded key-value pairs (..),
}
```

Figure 1: METADATA HTTP/2 frame

The METADATA frame defines the following flag:

END_METADATA (0x04): When set, the END_METADATA flag indicates that this frame ends the logical metadata block.

A METADATA frame without the END_METADATA flag set ***MUST*** be followed by a another METADATA frame on the same stream. However,

METADATA frames ***MAY*** be interleaved with non-METADATA frames on the same stream, or frames of any type on different streams.

METADATA frames are allowed on any stream. METADATA frames on stream 0 carry information pertaining to the whole connection. METADATA frames on any other stream are associated with the exchange carried by that stream.

METADATA frames do not alter the state of a stream. METADATA frames ***MUST NOT*** be sent on a stream in the "closed" or "half closed (local)" state. An endpoint that receives METADATA for a stream in the "idle" state ***MAY*** choose to retain the payload for a period of time, under the assumption that the stream will soon transition to the "open" state.

A metadata block is the concatenation of the payloads of a sequence of one or more METADATA frames, only the last of which has the END_METADATA flag set. If the transfer of the last metadata block cannot be completed due to the stream or connection being closed before a METADATA frame with the END_METADATA flag, then the incomplete metadata block ***SHOULD*** be discarded. This ***SHOULD NOT*** affect processing of previous metadata blocks on the same stream or connection.

METADATA frames obey the maximum frame size set by SETTINGS_MAX_FRAME_SIZE.

METADATA frames are not subject to flow control.

The metadata block of an HTTP/2 METADATA frame is encoded using HPACK instructions ([RFC7541]). An endpoint ***MUST NOT*** use any HPACK instructions that change the dynamic table.

3.2. METADATA HTTP/3 frame

The type of METADATA HTTP/3 frame is 0x4d.

```
METADATA HTTP/3 Frame {
  Type (i) = 0x4d,
  Length (i),

  Encoded key-value pairs (..),
}
```

Figure 2: METADATA HTTP/3 frame

METADATA frames are allowed on any stream that uses HTTP/3 frames. METADATA frames on the control stream carry information pertaining

to the whole connection. METADATA frames on a request stream or a push stream are associated with the exchange carried by that stream.

The metadata block of a HTTP/3 METADATA frame is encoded using QPACK representations. An endpoint ***MUST NOT*** use any QPACK representations that reference the dynamic table. Therefore the Required Insert Count is be zero, and decoding METADATA frame payloads do not elicit instructions on the QPACK decoder stream.

4. Negotiating METADATA

This document defines a new HTTP/2 setting identifier, SETTINGS_ENABLE_METADATA, with value 0x4d44. It also defines a new HTTP/3 setting identifier, SETTINGS_ENABLE_METADATA, with value 0x4d44.

An endpoint that supports METADATA frames ***SHOULD*** advertise that by sending SETTINGS_ENABLE_METADATA with value 1 on each connection. A value of 0 indicates that the endpoint does not support METADATA frames. A value other than 0 or 1 ***MUST NOT*** be sent. The initial value is 0. For HTTP/2, SETTINGS_ENABLE_METADATA ***MUST NOT*** be sent in any SETTINGS frame other than the first one.

An endpoint ***MAY*** send METADATA frames before it learns that the peer supports them. For example, a proxy might chose to forward METADATA frames, or it might chose to buffer them, before it receives a SETTINGS frame. An endpoint ***SHOULD NOT*** send METADATA frames after it learns that the peer does not support them.

5. Security Considerations

TODO Security

6. IANA Considerations

6.1. HTTP/2

This document adds an entry to the "HTTP/2 Frame Type" registry originally defined in [RFC7540] but updated to refer to [RFC9113] with the following parameters:

Code: 0x4d

Frame Type: METADATA

Reference: [[this document]]

This document adds an entry to the "HTTP/2 Settings" registry originally defined in [RFC7540] but updated to refer to [RFC9113] with the following parameters:

Code:
0x4d44

Name: SETTINGS_ENABLE_METADATA

Initial Value: 0

Reference: [[this document]]

6.2. HTTP/3

This document adds an entry to the "HTTP/3 Frame Types" registry defined in [RFC9114] with the following parameters:

Value: 0x4d

Frame Type: METADATA

Reference: [[this document]]

This document adds an entry to the "HTTP/3 Frame Types" registry defined in [RFC9114] with the following parameters:

Value: 0x4d44

Settings Name: SETTINGS_ENABLE_METADATA

Default: 0

Reference: [[this document]]

7. References

7.1. 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/rfc/rfc2119>.

[RFC7541] Peon, R. and H. Ruellan, "HPACK: Header Compression for HTTP/2", RFC 7541, DOI 10.17487/RFC7541, May 2015, <https://www.rfc-editor.org/rfc/rfc7541>.

[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/rfc/rfc8174>.

[RFC9113] Thomson, M., Ed. and C. Benfield, Ed., "HTTP/2", RFC 9113, DOI 10.17487/RFC9113, June 2022, <https://www.rfc-editor.org/rfc/rfc9113>.

[RFC9114] Bishop, M., Ed., "HTTP/3", RFC 9114, DOI 10.17487/RFC9114, June 2022, <https://www.rfc-editor.org/rfc/rfc9114>.

7.2. Informative References

[RFC7540] Belshe, M., Peon, R., and M. Thomson, Ed., "Hypertext Transfer Protocol Version 2 (HTTP/2)", RFC 7540, DOI 10.17487/RFC7540, May 2015, <https://www.rfc-editor.org/rfc/rfc7540>.

Acknowledgments

The authors would like to acknowledge Dianna Hu and Ian Swett for their contributions to this document.

Authors' Addresses

Bence Béky
Google LLC

Email: bnc@google.com

Biren Roy
Google LLC

Email: birenroy@google.com