Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: March 14, 2019

E. Kinnear
T. Pauly
Apple Inc.

September 10, 2018

Using HTTP/2 as a Transport for Arbitrary Bytestreams
draft-kinnear-httpbis-http2-transport-00

Abstract

HTTP/2 provides multiplexing of HTTP requests over a single underlying transport connection. HTTP/2 Transport defines a transport abstraction provided by HTTP/2 framing that is separate from HTTP semantics.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction 2
1.1. Notational Conventions 2
2. The STREAM HTTP/2 Frame 2
2.1. Syntax 3
2.2. Advertising Support for STREAM Frames 4
2.3. Processing STREAM Frames 4
3. IANA Considerations 5
4. Security Considerations 5
5. Acknowledgments 5
6. Normative References 5
Authors' Addresses 5

1. Introduction

HTTP/2 [RFC7540] provides a framing layer that describes the exchange of HTTP messages following HTTP semantics. This framing layer provides multiplexing of multiple streams on a single underlying transport connection, flow control, stream dependencies and priorities, and exchange of configuration information between endpoints. HTTP/2 also defines the mapping of HTTP semantics onto that framing layer.

This document defines the use of the HTTP/2 framing layer as a transport for arbitrary byte streams without the use of HTTP semantics.

1.1. Notational Conventions

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.

2. The STREAM HTTP/2 Frame

This document defines a new HTTP/2 frame type called STREAM, that allows endpoints to open HTTP/2 streams without header values. Either endpoint can send this frame to open a stream. STREAM frames are treated in all ways as HEADERS frames, including in the stream state machine, but are not required to contain any header values.

2.1. Syntax

The STREAM frame type is 0xd (decimal 13) and contains similar fields to that of the HEADERS frame.

A STREAM frame is shown below.

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +---------------+
   |Pad Length? (8)|
   +-+-------------+-----------------------------------------------+
   |E|                 Stream Dependency? (31)                     |
   +-+-------------+-----------------------------------------------+
   |  Weight? (8)  |
   +-+-------------+-----------------------------------------------+
   |                           Padding (*)                       ...
   +---------------------------------------------------------------+
```

Figure 1: STREAM Frame Format

The STREAM frame contains the following fields:

Pad Length: An 8-bit field containing the length of the frame padding in units of octets. This field is only present if the PADDED flag is set.

E: A single-bit flag indicating that the stream dependency is exclusive (see Section 5.3 of [RFC7540]). This field is only present if the PRIORITY flag is set.

Stream Dependency: A 31-bit stream identifier for the stream that this stream depends on (see Section 5.3 of [RFC7540]). This field is only present if the PRIORITY flag is set.

Weight: An unsigned 8-bit integer representing a priority weight for the stream (see Section 5.3 of [RFC7540]). Add one to the value to obtain a weight between 1 and 256. This field is only present if the PRIORITY flag is set.

Padding: Padding octets.

The STREAM frame defines the following flags:

PADDED (0x8): When set, bit 3 indicates that the Pad Length field and any padding that it describes are present.

PRIORITY (0x20): When set, bit 5 indicates that the Exclusive Flag (E), Stream Dependency, and Weight fields are present; see Section 5.3 of [RFC7540].

STREAM frames MUST be associated with a stream. If a STREAM frame is received whose stream identifier field is 0x0, the recipient MUST respond with a connection error of type PROTOCOL_ERROR.

The STREAM frame can include padding. Padding fields and flags are identical to those defined for DATA frames.

Prioritization information in a STREAM frame is logically equivalent to a separate PRIORITY frame, but inclusion in a STREAM frame avoids the potential for churn in stream prioritization when new streams are created. Prioritization fields in STREAM frames subsequent to the first on a stream reprioritize the stream (Section 5.3.3 of [RFC7540]).

2.2. Advertising Support for STREAM Frames

As defined in [RFC7540], both endpoints can send SETTINGS_MAX_CONCURRENT_STREAMS in SETTINGS frames to indicate the number of streams that the sender permits the receiver to create. This limit applies to streams created via the STREAM frame as well as streams created via HEADERS frames.

2.3. Processing STREAM Frames

The STREAM frame is a non-critical extension to HTTP/2. Endpoints that do not support this frame can safely ignore it upon receipt.

When received by a client that implements support, the STREAM frame behaves in the same manner as a HEADERS frame, but does not carry any header blocks. This changes the connection state in the same manner as a HEADERS frame, described in Section 4.3 of [RFC7540].

STREAM frames can be sent on a stream in the "idle", "reserved (local)", "open", or "half-closed (remote)" state. STREAM frames can be sent by either endpoint on a connection.

Streams created via a STREAM frame are multiplexed in the same manner on the underlying transport connection as streams created via a HEADERS frame. Flow control also applies to these streams in the same way. Flow control, stream dependencies, and priorities continue to apply to streams as defined by [RFC7540].

Anywhere an endpoint would be permitted to send a HEADERS frame by [RFC7540], it is likewise permitted to send a STREAM frame.

A stream is closed via a RST_STREAM frame or by setting the END_STREAM flag on a DATA frame.

3. IANA Considerations

This specification adds an entry to the "HTTP/2 Frame Type" registry.

- Frame Type: STREAM

- Code: 0xd

- Specification: [[RFC Editor: Please fill in this value with the RFC number for this document.]]

4. Security Considerations

5. Acknowledgments

Thanks to Anthony Chivetta, Joshua Otto, and Valentin Pistol for their contributions in the design and prototyping of this work.

6. 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>.

[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/info/rfc7540>.

[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>.

Authors' Addresses

Eric Kinnear
Apple Inc.
One Apple Park Way
Cupertino, California 95014
United States of America

Email: ekinnear@apple.com

Tommy Pauly
Apple Inc.
One Apple Park Way
Cupertino, California 95014
United States of America

Email: tpauly@apple.com