httpbis Working Group
Internet-Draft
Intended status: Standards Track
Expires: September 25, 2019

G. Xie
F. Frindell
Facebook Inc.

March 24, 2019

# An HTTP/2 extension for bidirectional messaging communication
## draft-xie-bidirectional-messaging-00

Abstract

This draft proposes a http2 protocol extension, which enables bidirectional messaging communication between client and server.

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 September 25, 2019.

1.

HTTP/2 is the de facto application protocol in Internet. The optimizations developed in HTTP/2, like stream multiplexing, header compression, and binary message framing are very generic. They can be useful in non web browsing applications, for example, Publish/ Subscribe, RPC. However, the request/response from client to server communication pattern limits HTTP/2 from wider use in these applications. This draft proposes a HTTP/2 protocol extension, which enables bidirectional messaging between client and server.

The only mechanism HTTP/2 provides for server to client communication is PUSH

PROMISE and the bi-directionality of HEADERS. Further, clients are also able group streams together for routing purposes, such that each individual stream does not need to carry additional routing information.

2.

The keywords

,

,

,

,

,

,

,

,

, and

, when they appear in this document, are to be interpreted as described in [RFC2119].

All the terms defined in the Conventions and Terminology section in [RFC7540] apply to this document.

3.

3.1.

A routing stream (RStream) is a long lived HTTP/2 stream in nature. RStreams are initiated by clients, and can be routed independently by any intermediaries. Though an RStream is effectively a regular HTTP/2 stream, RStreams are recommended for exchanging metadata, but not user data.

A new HTTP/2 stream called ExStream is introduced for exchanging user data. ExStreams are recommended for short lived transactions, so intermediaries and servers can gracefully shutdown ExStreams within a short time. The typical use case can be a subscription or publish request/response in Publish/Subscribe use case, or an RPC call between two endpoints.

An ExStream is opened by an EX_HEADERS frame, and continued by CONTINUATION and DATA frames. An ExStream

be associated with an open RStream, and

be associated with any other ExStream. ExStreams are routed according to their RStreams by intermediaries and servers. Effectively, all ExStreams with the same RStream form a logical stream group, and are routed to the same endpoint.

3.2.

With RStreams and ExStreams, HTTP/2 can be used for bidirectional messaging communication. As shown in the follow diagrams, after an RStream is open from client to server, either endpoint can initiate an ExStreams to its peer.

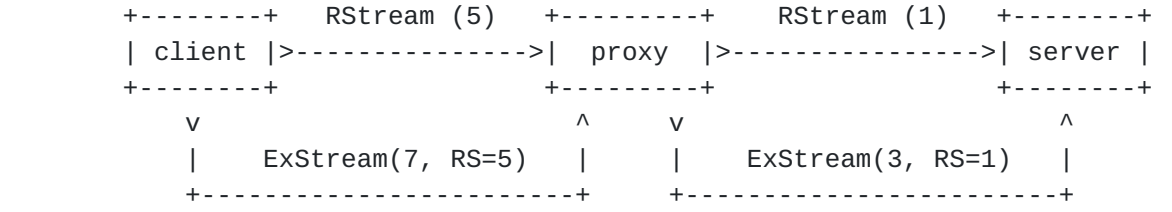


Figure 1

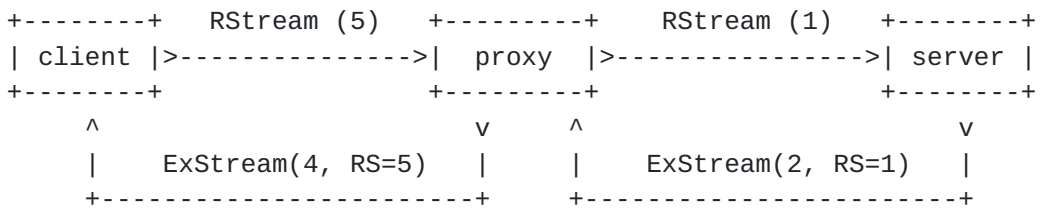


Figure 2

Beyond that, clients can multiplex RStreams, ExStreams and regular HTTP/2 streams into one HTTP/2 connection. This enables clients to access different services without initiating new TCP connections. This avoids the latency cost of setting up new connections. This is more desirable for mobile devices because they usually have longer RTT and battery constraints. Multiplexing these services also allows them to share a single TCP connection congestion control context.

As shown in the following diagram, the client can exchange data with PubSub, RPC and CDN three different services with one TCP connection.

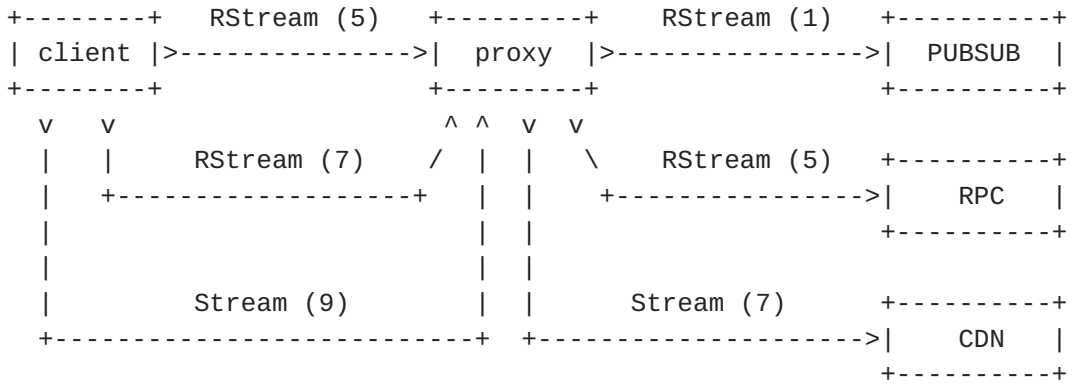


Figure 3

3.3.

RStreams are regular HTTP/2 streams that follow the stream lifecycle in [RFC7540], section 5.1. ExStreams use the same lifecycle as regular HTTP/2 streams, but also depend on their RStreams. If a RStream is reset, endpoints

reset the ExStreams associated with that RStream. If the RStream is closed, endpoints SHOULD allow existing ExStreams complete normally. The RStream SHOULD remain open while communication is ongoing. Endpoints SHOULD refresh any timeouts on the RStream while associated ExStreams are open.

A sender

initiate new ExStreams if on an RStream that is in the open or half closed (remote) state.

Endpoints process new ExStreams only when the RStream is open or half closed (local) state. If an endpoint receives an EX

STREAM_ERROR.

3.4.

The extension SHOULD be disabled by default. Endpoints can negotiate the use of the extension through the SETTINGS frame. If an implementation supports the extension, it is RECOMMENDED to include the ENABLE

HEADERS setting in the initial SETTINGS frame. HTTP/2 compliant implementations will ignore the setting if it is unknown. An endpoint can send EX

EX_HEADERS=1.

Endpoints MUST NOT send out EX

EX

HEADERS will be ignored, making the header compression contexts inconsistent between sender and receiver.

If an endpoint supports this extension, but receives EX

EX

HEADER

ENABLED_ERROR.

Intermediaries SHOULD send the ENABLE

HEADERS setting to clients, only if intermediaries and their upstream servers can support this extension. If an intermediary receives an ExStream but discovers the destination endpoint does not support the extension, it MUST reset the stream with EX

NOT

ERROR.

3.5.

The extension implementation should apply stream and connection level flow control, maximum concurrent streams limit, GOAWAY logic to both RStreams and ExStreams.

4.

The EX

HEADERS has one extra field, RStream ID. It is used to open an ExStream, and additionally carries a header block fragment. EX_HEADERS frames can be sent on a stream in the "idle", "open", or "half-closed (remote)" state.

Like HEADERS, the CONTINUATION frame (type=0x9) is used to continue a sequence of header block fragments, if the headers do not fit into one EX_HEADERS frame.

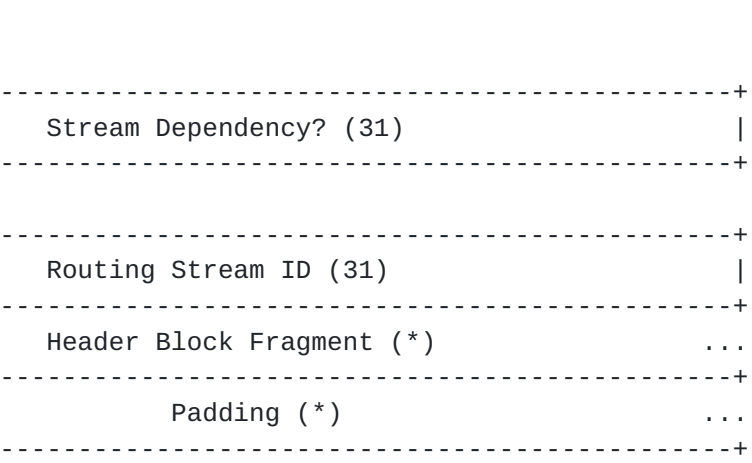


Figure 4

The RStream specified in EX

STREAM

ERROR, if the specified RStream is missing; or is an ExStream rather than a stream; or is closed or half-closed (remote). Otherwise, the states maintained for header compression or flow control) may be out of sync.

5.

This document establishes a registry for a new frame type, setting, and error code.

5.1.

The entry in the following table are registered by this document.

5.2.

The entry in the following table are registered by this document.

5.3.

The entry in the following table are registered by this document.

## 6. References

Authors' Addresses

Guowu Xie
Facebook Inc.
1 Hacker Way
Menlo Park CA 94025
U.S.A.

Email: woo@fb.com

Alan Frindell
Facebook Inc.

Email: afrind@fb.com