Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: May 5, 2020

V. Vasiliev
Google

November 2, 2019

WebTransport over HTTP/3
draft-vvv-webtransport-http3-01

Abstract

WebTransport [OVERVIEW] is a protocol framework that enables clients constrained by the Web security model to communicate with a remote server using a secure multiplexed transport. This document describes Http3Transport, a WebTransport protocol that is based on HTTP/3 [HTTP3] and provides support for unidirectional streams, bidirectional streams and datagrams, all multiplexed within the same HTTP/3 connection.

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 May 5, 2020.

Table of Contents

1. Introduction

HTTP/3 [HTTP3] is a protocol defined on top of QUIC [QUIC-TRANSPORT] that can multiplex HTTP requests over a QUIC connection. This document defines Http3Transport, a mechanism for multiplexing non-HTTP data with HTTP/3 in a manner that conforms with the WebTransport protocol framework [OVERVIEW]. Using the mechanism described here, multiple Http3Transport instances can be multiplexed simultaneously with regular HTTP traffic on the same HTTP/3 connection.

1.1. Terminology

The keywords "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.

This document follows terminology defined in Section 1.2 of [OVERVIEW]. Note that this document distinguishes between a WebTransport server and an HTTP/3 server. An HTTP/3 server is the server that terminates HTTP/3 connections; a WebTransport server is an application that accepts WebTransport sessions, which can be accessed via an HTTP/3 server.

2. Protocol Overview

Http3Transport servers are identified by a pair of authority value and path value (defined in [RFC3986] Sections 3.2 and 3.3 correspondingly).

When an HTTP/3 connection is established, the client and server have to negotiate a specific set of QUIC transport parameters that indicate support for various Http3Transport features. Most notably, the "http3_transport_support" parameter signals Http3Transport support to the peer.

Http3Transport sessions are initiated inside a given HTTP/3 connection by the client, who sends an extended CONNECT request [RFC8441]. If the server accepts the request, an Http3Transport session is established. As a part of this process, the client proposes, and the server confirms, a session ID. A session ID (SID) is unique within a given HTTP/3 connection, and is used to associate all of the streams and datagrams with the specific session.

After the session is established, the peers can exchange data using the following mechanisms:

- A client can create a bidirectional stream using a special indefinite-length HTTP/3 frame that transfers ownership of the stream to Http3Transport.

- A server can create a bidirectional stream, which is possible since HTTP/3 does not define any semantics for server-initiated bidirectional streams.

- Both client and server can create a unidirectional stream using a special stream type.

- A datagram can be sent using a QUIC DATAGRAM frame [QUIC-DATAGRAM].

An Http3Transport session is terminated when the CONNECT stream that created it is closed.

3. Session IDs

In order to allow multiple Http3Transport sessions to occur within the same HTTP/3 connection, Http3Transport assigns every session a unique ID, further referred to as session ID. A session ID is a 62-bit number that is unique within the scope of an HTTP/3 connection, and is never reused even after the session is closed. The client unilaterally picks the session ID. As the IDs are encoded using QUIC variable length integers, the client SHOULD start with zero and then sequentially increment the IDs. A session ID is considered to be used, and thus ineligible for new transports, as soon as the client sends a request proposing it. These reuse requirements guarantee that both HTTP/3 endpoints have a consistent view of the session ID space.

The Session ID is a hop-by-hop property: if Http3Transport is proxied, the same session can have different IDs from the client's and server's perspective. Because of that, session IDs SHOULD NOT be exposed to the application.

4. Session Establishment

4.1. Establishing a Transport-Capable HTTP/3 Connection

In order to indicate support for Http3Transport, both the client and server MUST send an empty "http3_transport_support" transport parameter. Endpoints MUST NOT use any Http3Transport-related functionality unless the parameter has been negotiated. The negotiation is done through a QUIC transport parameter instead of an HTTP/3-level setting as it allows the server to customize the transport parameters it intends to send based on whether the client has indicated support for Http3Transport.

If "http3_transport_support" is negotiated, support for the QUIC DATAGRAM extension MUST be negotiated. The "initial_max_bidi_streams" MUST be greater than zero, overriding the existing requirement in [HTTP3].

4.2. Extended CONNECT in HTTP/3

[RFC8441] defines an extended CONNECT method in Section 4, enabled by the SETTINGS_ENABLE_CONNECT_PROTOCOL parameter. That parameter is only defined for HTTP/2. This document does not create a new multi-purpose parameter to indicate support for extended CONNECT in HTTP/3;

instead, the "http3_transport_support" transport parameter implies that an endpoint supports extended CONNECT.

4.3. Creating a New Session

As Http3Transport sessions are established over HTTP/3, they are identified using the "https" URI scheme [RFC7230].

In order to create a new Http3Transport session, a client can send an HTTP CONNECT request. The ":protocol" pseudo-header field MUST be set to "webtransport". The ":scheme" field MUST be "https". Both the ":authority" and the ":path" value MUST be set; those fields indicate the desired WebTransport server. The client MUST pick a new session ID as described in Section 3 and send it encoded as a hexadecimal literal in ":sessionid" header. An "Origin" header [RFC6454] MUST be provided within the request.

Upon receiving an extended CONNECT request with a ":protocol" field set to ":webtransport", the HTTP/3 server can check if it has a WebTransport server associated with the specified ":authority" and ":path" values. If it does not, it SHOULD reply with status code 404 (Section 6.5.4, [RFC7231]). If it does, it MAY accept the session by replying with status code 200. Before accepting it, the HTTP/3 server MUST verify that the proposed session ID does not conflict with any currently open sessions, and it MAY verify that it was not used ever before on this connection. The WebTransport server MUST verify the "Origin" header to ensure that the specified origin is allowed to access the server in question.

From the client's perspective, an Http3Transport session is established when the client receives a 200 response. From the server's perspective, a session is established once it sends a 200 response. Both endpoints MUST NOT open any streams or send any datagrams on a given session before that session is established. Http3Transport does not support 0-RTT.

4.4. Limiting the Number of Simultaneous Sessions

From the flow control perspective, Http3Transport sessions count against the stream flow control just like regular HTTP requests, since they are established via an HTTP CONNECT request. This document does not make any effort to introduce a separate flow control mechanism for sessions, nor to separate HTTP requests from WebTransport data streams. If the server needs to limit the rate of incoming requests, it has alternative mechanisms at its disposal:

- "HTTP_REQUEST_REJECTED" error code defined in [HTTP3] indicates to the receiving HTTP/3 stack that the request was not processed in any way.

- HTTP status code 429 indicates that the request was rejected due to rate limiting [RFC6585]. Unlike the previous method, this signal is directly propagated to the application.

5. WebTransport Features

Http3Transport provides the following features described in [OVERVIEW]: unidirectional streams, bidirectional streams and datagrams, initiated by either endpoint.

Session IDs are used to demultiplex streams and datagrams belonging to different Http3Transport sessions. On the wire, session IDs are encoded using the QUIC variable length integer scheme described in [QUIC-TRANSPORT].

5.1. Unidirectional streams

Once established, both endpoints can open unidirectional streams. The HTTP/3 control stream type SHALL be 0x54. The body of the stream SHALL be the stream type, followed by the session ID, encoded as a variable-length integer, followed by the user-specified stream data (Figure 1).

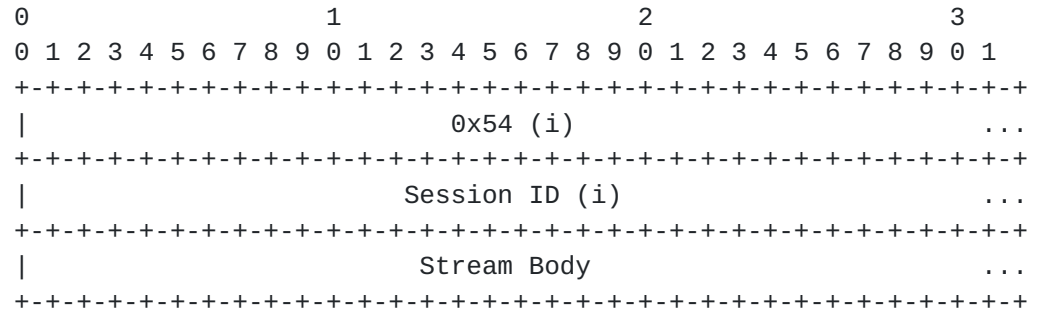

Figure 1: Unidirectional Http3Transport stream format

5.2. Client-Initiated Bidirectional Streams

Http3Transport clients can initiate bidirectional streams by opening an HTTP/3 bidirectional stream and sending an HTTP/3 frame with type "WEBTRANSPORT_STREAM" (type=0x41). The format of the frame SHALL be the frame type, followed by the session ID, encoded as a variable-length integer, followed by the user-specified stream data (Figure 2). The frame SHALL last until the end of the stream.

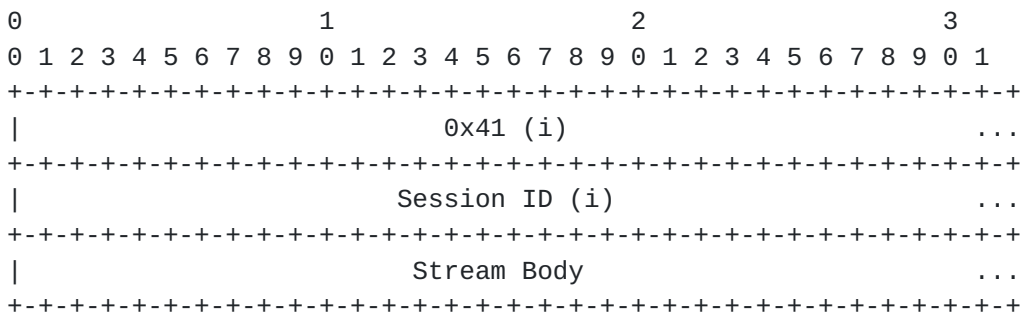

Figure 2: WEBTRANSPORT_STREAM frame format

5.3. Server-Initiated Bidirectional Streams

Http3Transport servers can initiate bidirectional streams by opening a bidirectional stream within the HTTP/3 connection. Note that since HTTP/3 does not define any semantics for server-initiated bidirectional streams, this document is a normative reference for the semantics of such streams for all HTTP/3 connections in which the "http3_transport_support" option is negotiated. The format of those streams SHALL be the session ID, encoded as a variable-length integer, followed by the user-specified stream data (Figure 3).

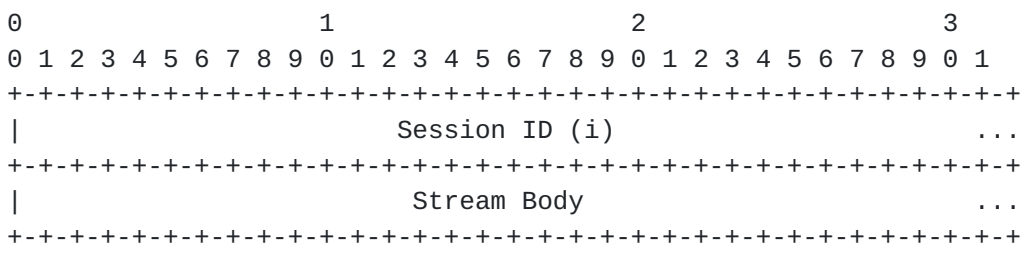

Figure 3: Server-initiated bidirectional stream format

5.4. Datagrams

Datagrams can be sent using the DATAGRAM frame as defined in [QUIC-DATAGRAM] and [H3-DATAGRAM]. For all HTTP/3 connections in which the "http3_transport_support" option is negotiated, the Flow Identifier is set to the session ID. In other words, the format of datagrams SHALL be the session ID, followed by the user-specified payload (Figure 4).

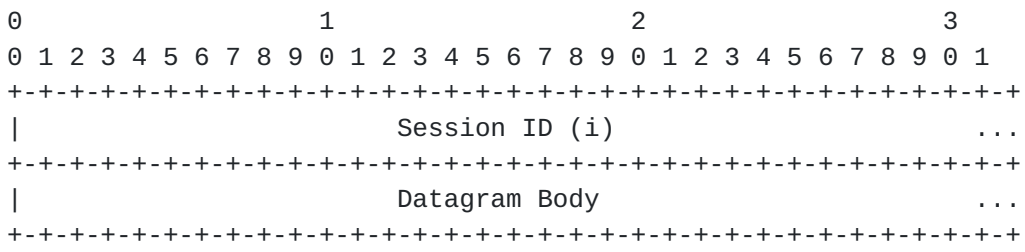

Figure 4: Datagram format

In QUIC, a datagram frame can span at most one packet. Because of that, the applications have to know the maximum size of the datagram they can send. However, when proxying the datagrams, the hop-by-hop MTUs can vary. TODO: Describe how the path MTU can be computed, specifically propagation across HTTP proxies.

6. Session Termination

An Http3Transport is terminated when either endpoint closes the stream associated with the CONNECT request that initiated the session. Upon learning about the session being terminated, the endpoint MUST stop sending new datagrams and reset all of the streams associated with the session.

7. Transport Properties

Http3Transport supports most of the WebTransport features described in Table 1.

Property	Support
Stream independence	Always supported
Partial reliability	Always supported
Pooling support	Always supported
Connection mobility	Implementation-dependent

Table 1: Transport properties of Http3Transport

8. Security Considerations

Http3Transport satisfies all of the security requirements imposed by [QUIC-TRANSPORT] on WebTransport protocols, thus providing a secure framework for client-server communication in cases when the client is

potentially untrusted. Since HTTP/3 is QUIC-based, a lot of the analysis in [WEBTRANSPORT-QUIC] applies here.

Http3Transport requires explicit opt-in through the use of a QUIC transport parameter; this avoids potential protocol confusion attacks by ensuring the HTTP/3 server explicitly supports it. It also requires the use of the Origin header, providing the server with the ability to deny access to Web-based clients that do not originate from a trusted origin.

Just like HTTP/3 itself, Http3Transport pools traffic to different origins within a single connection. Different origins imply different trust domains, meaning that the implementations have to treat each transport as potentially hostile towards others on the same connection. One potential attack is a resource exhaustion attack: since all of the transports share both congestion control and flow control context, a single client aggressively using up those resources can cause other transports to stall. The user agent thus SHOULD implement a fairness scheme that ensures that each transport within connection gets a reasonable share of controlled resources; this applies both to sending data and to opening new streams.

9. IANA Considerations

9.1. Upgrade Token Registration

The following entry is added to the "Hypertext Transfer Protocol (HTTP) Upgrade Token Registry" registry established by [RFC7230]:

The "webtransport" label identifies HTTP/3 used as a protocol for WebTransport:

Value: webtransport

Description: WebTransport over HTTP/3

Reference: This document

9.2. QUIC Transport Parameter Registration

The following entry is added to the "QUIC Transport Parameter Registry" registry established by [QUIC-TRANSPORT]:

The "http3_transport_support" parameter indicates that the specified HTTP/3 connection is Http3Transport-capable.

Value: 0x????

Parameter Name: http3_transport_support

Specification: This document

9.3. Frame Type Registration

The following entry is added to the "HTTP/3 Frame Type" registry established by [HTTP3]:

The "WEBTRANSPORT_STREAM" frame allows HTTP/3 client-initiated bidirectional streams to be used by WebTransport:

Code: 0x54

Frame Type: WEBTRANSPORT_STREAM

Specification: This document

9.4. Stream Type Registration

The following entry is added to the "HTTP/3 Stream Type" registry established by [HTTP3]:

The "WebTransport stream" type allows unidirectional streams to be used by WebTransport:

Code: 0x41

Stream Type: WebTransport stream

Specification: This document

Sender: Both

10. References

10.1. Normative References

[H3-DATAGRAM] Schinazi, D., "Using QUIC Datagrams with HTTP/3", draft-schinazi-quic-h3-datagram-01 (work in progress), October 2019.

[HTTP3] Bishop, M., Ed., "Hypertext Transfer Protocol Version 3 (HTTP/3)", draft-ietf-quic-http (work in progress).

[OVERVIEW]
Vasiliev, V., "The WebTransport Protocol Framework", draft-vvv-webtransport-overview-01 (work in progress).

[QUIC-DATAGRAM]
Pauly, T., Kinnear, E., and D. Schinazi, "An Unreliable Datagram Extension to QUIC", draft-pauly-quic-datagram (work in progress).

[QUIC-TRANSPORT]
Iyengar, J., Ed. and M. Thomson, Ed., "QUIC: A UDP-Based Multiplexed and Secure Transport", draft-ietf-quic-transport (work in progress).

[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>.

[RFC3986] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform Resource Identifier (URI): Generic Syntax", STD 66, RFC 3986, DOI 10.17487/RFC3986, January 2005, <https://www.rfc-editor.org/info/rfc3986>.

[RFC6454] Barth, A., "The Web Origin Concept", RFC 6454, DOI 10.17487/RFC6454, December 2011, <https://www.rfc-editor.org/info/rfc6454>.

[RFC6585] Nottingham, M. and R. Fielding, "Additional HTTP Status Codes", RFC 6585, DOI 10.17487/RFC6585, April 2012, <https://www.rfc-editor.org/info/rfc6585>.

[RFC7230] Fielding, R., Ed. and J. Reschke, Ed., "Hypertext Transfer Protocol (HTTP/1.1): Message Syntax and Routing", RFC 7230, DOI 10.17487/RFC7230, June 2014, <https://www.rfc-editor.org/info/rfc7230>.

[RFC7231] Fielding, R., Ed. and J. Reschke, Ed., "Hypertext Transfer Protocol (HTTP/1.1): Semantics and Content", RFC 7231, DOI 10.17487/RFC7231, June 2014, <https://www.rfc-editor.org/info/rfc7231>.

[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>.

[RFC8441] McManus, P., "Bootstrapping WebSockets with HTTP/2", RFC 8441, DOI 10.17487/RFC8441, September 2018, <https://www.rfc-editor.org/info/rfc8441>.

10.2. Informative References

[WEBTRANSPORT-QUIC]
Vasiliev, V., "WebTransport over QUIC", draft-vvv-webtransport-quic-01 (work in progress).

Author's Address

Victor Vasiliev
Google

Email: vasilvv@google.com